TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Page

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS AND ACRONYMS

AI	all-intra
AVC	advanced video coding
BD-Rate	Bjøntegaard delta rate
BD-PSNR	Bjøntegaard delta peak signal-to-noise ratio
BIC	Bayesian information criterion
CB	coding block
CDF	cumulative distribution function
CL	confidence level
CTB	coding tree block
CTC	common test conditions
CTU	coding tree unit
CU	coding unit
DP	dynamic programming
HD	high-definition
HEVC	high efficiency video coding
HM	HEVC test model
JCT-VC	joint collaborative team on video coding
JM	joint model

LD	low delay
MDGA	mean of directional gradient amplitudes
MDP	markov decision process
MGA	mean of gradient amplitudes
MI	mutual information
MLP	multi-layer perceptron
MPEG	moving picture experts group
MPM	most probable mode
MRM	most relevant mode
MSE	mean squared error
PB	prediction block
PDF	probability distribution function
PSNR	peak signal to noise ratio
PU	prediction unit
QP	quantization parameter
RA	random access
RD	rate-distortion
RDO	rate-distortion optimization
RL	reinforcement learning
RMD	rough mode decision

RQT	residual quad-tree
SAD	sum of absolute differences
SATD	sum of absolute transformed differences
SSE	sum of squared errors
SVM	support vector machine
TB	transform block
TR	time reduction
TU	transform unit
VCEG	video coding experts group

LISTE OF SYMBOLS AND UNITS OF MEASUREMENTS

ω_c	Chroma weight for RDO cost computation.
R_m	Number of bits for PU encoding.
R_p	Number of bits used for mode coding.
λ_p	Lagrange multiplier in RMD cost.
λ_m	Lagrange multiplier in RDO cost.
α	Distance factor in mode classification.
B_{Mode}	Number of bits for signaling the best intra mode.
B_{Coding}	Number of bits for the encoded block after entropy coding.
γ	Discount factor in reinforcement learning framework.
$\mathscr{F}$	Set of transition samples in batch mode learning.
$\mathscr{T}$	Training set in batch mode learning.
ϕ	Mapping form state space to feature space.
$\hat{\pi}^*$	Estimation of an optimal policy.
ρ	Correlation coefficient.

CHAPTER 1

INTRODUCTION

The use of high quality video such as high-definition (HD) and beyond HD ($4K \times 2K$ and $8K \times 4K$) has created a demand for new video coding technologies beyond the capability of the well-known H.264/advanced video coding (AVC) standard (Wiegand *et al.*, 2003). In view of this, ITU-T video coding experts group (VCEG) and ISO/IEC moving picture experts group (MPEG) have established a joint group named the joint collaborative team on video coding (JCT-VC) to design a new video coding standard, called high efficiency video coding (HEVC)/H.265 (Sullivan *et al.*, 2012) in order to improve the coding efficiency as compared to the other video coding standards. It has been shown in (Ohm *et al.*, 2012) that HEVC provides a significant bit rate reduction, i.e., up to 50%, for the same subjective quality in comparison to H.264/AVC.

The improved coding performance of HEVC is achieved through a highly flexible coding structure using several advanced tools. For instance, in intra prediction, unlike the 9 modes used in H.264/AVC, the number of modes increases to 35, including dc, planar and 33 directional modes. This allows HEVC encoder to make predictions more precisely, and thus, to exploit the spatial correlation efficiently. In addition, HEVC provides a new content-adaptive and recursive approach to split a frame into coding units (CUs) in a quadtree-based manner, resulting in an improved coding efficiency. However, finding the best candidate among the 35 modes and the best splitting pattern requires a highly complex rate-distortion optimization (RDO) process. In view of this, the computational complexity and encoding time significantly increase; making the real-time coding challenging.

1.1 Problem Statement and Motivations

The coding efficiency improvement of HEVC comes along with higher computational complexity in the structure of the codec, which limits HEVC utilization in low-delay real-time applications. This complexity may be manageable at the decoder, yet quickly goes beyond

the capability of the current hardware at the encoder. Thus, new algorithms are needed to reduce the complexity of the encoder without sacrificing its coding performance, especially in designing video encoders for rapidly developing portable devices.

In light of this, in this thesis, we aim at reducing the HEVC encoder's computational complexity while preserving its high compression efficiency. The computational complexity of the encoder is studied and novel methods for complexity reduction are proposed. To this end, the focus is on coding mode decision and CU size decision, since they represent the most time-consuming modules of the encoder. The intra coding is selected because there is an *all-intra* profile for HEVC that is expected to replace the current intra coding techniques. In addition to *all-intra* profile, the intra coding is tested for all blocks even if a block is coded with an inter mode. This is due to the fact that for every block the encoder chooses between inter mode and intra mode. It is noted that the proposed methods may also be used in transcoding and transrating systems. Accordingly, the results of this thesis will help designing new generations of HEVC encoders, transcoders and transraters, especially in real-time applications.

1.2 Objectives

The objectives of this study are presented as follows:

- Computational complexity reduction of the HEVC intra encoder
- Design of fast encoding algorithms in HEVC intra coding
- Mathematical analysis and creation of predictive models based on experimental data for mode decision
- RDO cost modeling and prediction to avoid running RDO process in an exhaustive manner for all modes and all depths
- Formulation of the coding unit splitting problem as a learning problem to predict the splitting pattern based on block's features to do fast CU size decision

- Implementation of the proposed methods into the HEVC test model (Il-Koo, 2014) (HM)
- Implementation of the proposed methods into the highly optimized x265 HEVC encoder to transfer technology to industrial partner
- Publication of research results in reputable conferences and journals

The first objective is achieved by proposing techniques and methods for complexity reduction in the most time-consuming modules of the encoder. For instance, in mode decision, new algorithms are proposed to efficiently select the best mode among all candidates without losing the quality. In addition, in CU splitting, new methods are developed to find the best splitting pattern without performing exhaustive search for all coding levels. Since for every block the encoder chooses between inter mode and intra mode, the intra procedure is performed for all the blocks even if a block is coded in inter mode.

Since the high complexity encoder is usually run by the video providers and the decoder is in the hand-held devices, the objectives of this thesis help the video manufacturers provide high quality video content at a reasonable computational complexity. Reducing the complexity and encoding time allows the video providers to produce and deploy high quality and real-time applications with less hardware, which enables them to supply inexpensive video contents to consumers. In view of this, video industries will benefit from contributions of this thesis in both entertainment and interactive video applications as the main areas of video coding.

1.3 Contributions

In this thesis, several innovations are presented contributing to a significant complexity reduction in HEVC intra coding, while maintaining quality. The contributions are listed as follows:

- An RDO cost statistical modeling is developed for excluding non-promising candidate modes from further processing by rate-distortion optimization. More specifically, a

Gaussian modeling for the RDO cost is considered resulting in a very low-complexity and high-quality intra mode decision. This is due to the fact that it can effectively identify the smallest possible number of intra modes which need to be investigated by the rate-distortion optimization.

- Candidate selection is performed based on the bivariate normal distribution for RDO costs via considering correlation among these distributions.
- A chroma mode classification is carried out for fast chroma mode decision and a performance analysis of different proposed fast chroma mode decision approaches is conducted.
- Directional modes are selected by applying various gradient operators, and a performance analysis of different gradient kernels are conducted. An enhanced gradient-based mode decision is further proposed by assigning weights to three adjacent modes in order to improve the accuracy of the method.
- The neighboring blocks' best modes are used in a novel way to reduce the number of final modes that should be considered by the RDO process for the current block.
- A sum of absolute transformed differences (SATD) cost classification approach is developed to exclude non-promising candidates from RDO computations.
- New features are proposed for CU early splitting and CU early splitting termination.
- A new fast intra coding method is proposed based on global and directional gradients to early terminate the current CU splitting and avoid performing the high-complexity RDO process for the next CU levels.
- The CU size decision problem is addressed with two Bayesian classifiers for early splitting and early splitting termination.
- A new framework for CU size decision problem is proposed based on reinforcement learning, active feature acquisition and neural networks.

1.4 Thesis Structure

A brief overview of the HEVC video encoder is presented in chapter 2 in order to introduce the context and background of the thesis. We focus mainly on intra coding as the main subject of this study. The structure of the HEVC encoder is explained and it is shown how the new encoder achieves very high compression ratios.

The state-of-the-art studies that have been carried out during and after the standardization process of the HEVC are also presented. These studies have mostly focused on different encoder modules and propose methods for decreasing the complexity, while preserving the compression efficiency. These works may be categorized into two groups, namely, intra mode decision and coding unit size decision, reviewed in chapter 3.

In chapter 4, we present the proposed mode decision methods. To lower the high computational complexity of the HEVC encoder, in section 4.1, an RDO cost prediction method is developed based on statistical models, which selects only promising candidates for RDO. In section 4.2, a fast chroma mode decision, which results in an extra complexity reduction, is proposed to reduce the number of RDO candidates by applying a mode classification. Section 4.3 introduces a gradient-based approach, to exclude non-promising directional candidates from further processing. Section 4.4 discusses the SATD cost classification applied to the mode decision. The results presented in chapter 4 have partly been presented in the following papers:

- (Jamali and Coulombe, 2018). "Fast HEVC Intra Mode Decision Based on RDO Cost Prediction". Accepted in IEEE Transactions on Broadcasting.
- (Jamali and Coulombe, 2016a). "RDO Cost Modeling for Low-complexity HEVC Intra Coding". In IEEE Canadian Conference on Electrical and Computer Engineering (CCECE).
- (Jamali *et al.*, 2015). "Fast HEVC Intra Mode Decision Based on Edge Detection and SATD Costs Classification". In IEEE Data Compression Conference (DCC).

In addition, the following patent has been filed, which describes the proposed mode decision approach for complexity reduction in HEVC intra coding:

- (Jamali *et al.*, 2014). "Method and System for Fast Mode Decision for High Efficiency Video Coding". https://patents.google.com/patent/US20160127725A1/en.

In chapter 5, the proposed methods for CU size decision are presented. Section 5.1 presents a method for fast CU size decision based on global and directional gradients and section 5.2 introduces a Bayesian problem for early splitting and early splitting termination. Finally, a reinforcement learning-based approach is presented in section 5.3 for fast CU size decision. Part of the results provided in this chapter have also been presented in:

- (Jamali and Coulombe, 2016b). "Coding Unit Splitting Early Termination for Fast HEVC Intra Coding Based on Global and Directional Gradients". In IEEE Workshop on Multimedia Signal Processing (MMSP).

In section 5.3.4, the experimental results are provided and finally the conclusion of the thesis and discussion on the future directions of the research in HEVC complexity reduction are presented.

CHAPTER 2

HEVC VIDEO ENCODER OVERVIEW

In this chapter, we give a review on HEVC video encoding while focusing on intra coding. HEVC, similar to previous video coding standards, is a hybrid video coding scheme with motion estimation and compensation for decreasing the temporal redundancy, and intra prediction and transform for decreasing the spatial redundancy. Also the last step in the encoding process is an entropy coding module for further compression. Although the main functions in HEVC are similar to prior video coding standards, most of them have evolved compared to those standards. The improvements are mainly in frame splitting into prediction and transform blocks, inter and intra mode decision, in-loop filtering and entropy coding. Before we describe the structure of the HEVC encoder, we give a brief review on general hybrid video coding to better understand the context of the study.

2.1 Block-Based Hybrid Video Coding

The compression efficiency of video encoders has greatly improved during last three decades. However, the main concepts of video coding are still the same. All main video coding standards have been based on a same concept of block-based hybrid video coding (Richardson, 2010). Hybrid video coding is based on redundancy removal in some consecutive steps. The term hybrid means using both predictive coding and transform coding. Figure 2.1 shows a simplified structure of a hybrid video encoder. Each frame is partitioned into some rectangular regions called blocks which are coded by either inter or intra prediction. Inter prediction uses the previous frames to find the best prediction for the current block while intra prediction uses the information available in the current frame to make the prediction. The difference between the current block and its prediction forms the residual block. A transform is applied to the residual to remove spatial redundancy and to provide a more appropriate representation of the residual block for quantization. The quantization step provides a trade-off between compression ratio

and visual quality. Then entropy coding is used as the final step to achieve the maximum possible compression.

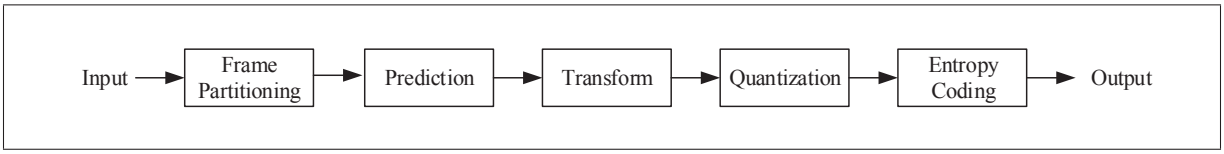

Figure 2.1 General view of a hybrid video encoder.

2.2 HEVC Video Encoding

The architecture of HEVC is basically similar to the previous hybrid video coding standards such as H.264/AVC. Figure 2.2 shows the structure of the HEVC encoder. The most important difference between HEVC and H.264/AVC is how the splitting of a frame into blocks is performed for achieving prediction and transform, which contributes to an improved coding efficiency. While the main coding block in H.264/AVC is a macroblock with size of 16×16, HEVC uses a more adaptive quadtree structure based on a block called coding tree unit (CTU) with maximum size of 64×64. This quadtree structure comprises some blocks and units. A block includes a rectangular area of samples and a unit is formed by a luma block and two related chroma blocks with associated syntax information. For example, a CTU consists of a luma coding tree block (CTB) and two chroma CTBs with syntax determining the further subdivision. As a result of subdivision, new units called CU are generated. The CU is a unit which is coded by inter or intra prediction. The CUs could be divided into prediction units (PUs) and transform units (TUs) for performing prediction and transform. PUs inside a CU can be predicted by different prediction modes. CUs, PUs and TUs consist of associated luma and chroma blocks called coding blocks (CBs), prediction blocks (PBs) and transform blocks (TBs) respectively. As it is obvious, this splitting is more adaptive relative to the approach used in H.264/AVC and is especially useful for higher resolution videos like $4K \times 2K$ and $8K \times 4K$. Figure 2.3 shows an example of dividing a frame into CUs and PUs using quadtree structure and fig. 2.4 shows the coding order of CUs inside a CTU. The quadtree that is used for dividing

a CU into TUs has its root at the CU level and is called residual quad-tree (RQT) because it operates on the residual. Also, all possible modes of splitting a CU into PUs are illustrated in fig. 2.5.

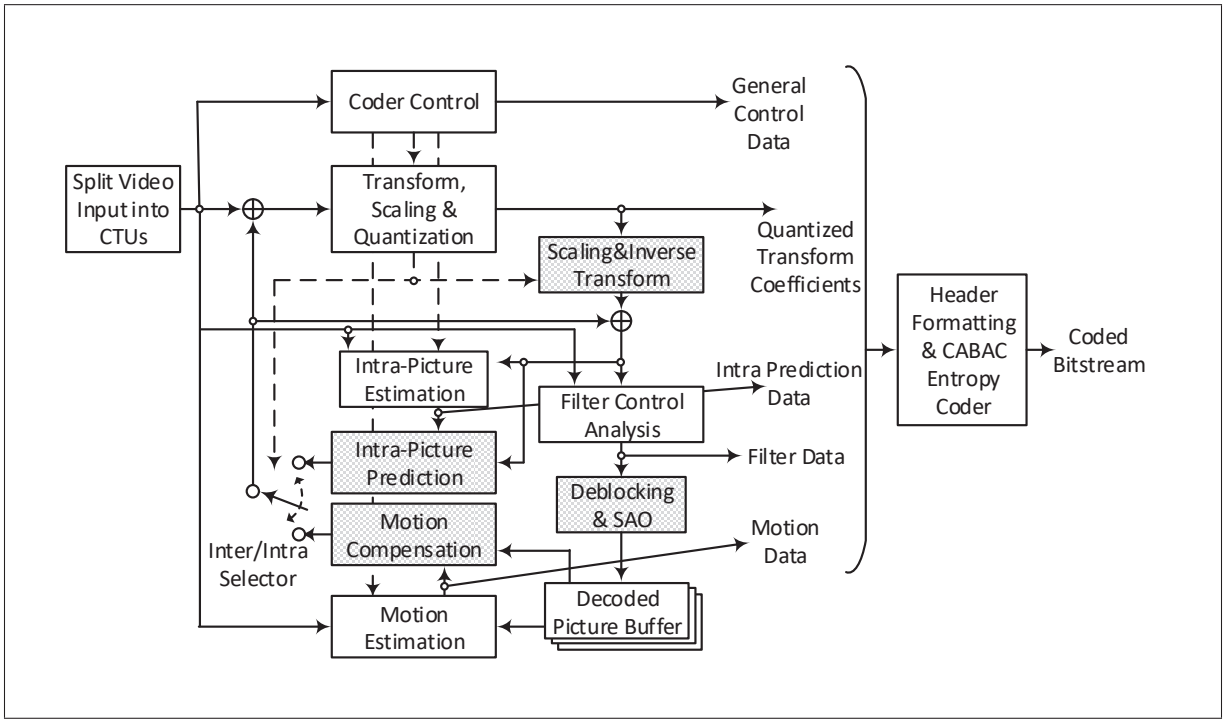

Figure 2.2 HEVC video encoder, (Hojati, 2018).

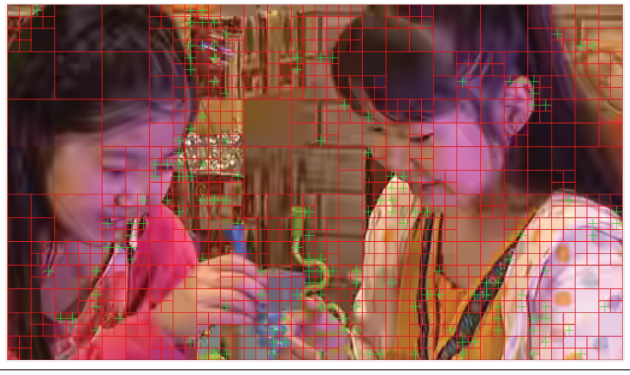

Figure 2.3 Frame splitting into CUs (red) and PUs (green) based on the quadtree structure, BlowingBubbles.

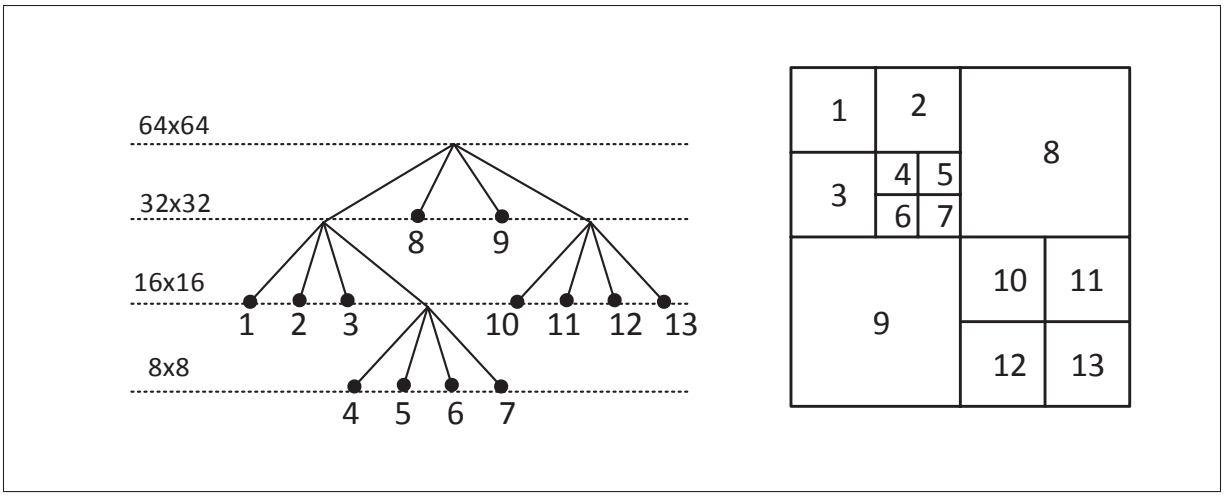

Figure 2.4 Partitioning of a 64×64 CTU into CUs and the coding order of CUs, (Hojati, 2018).

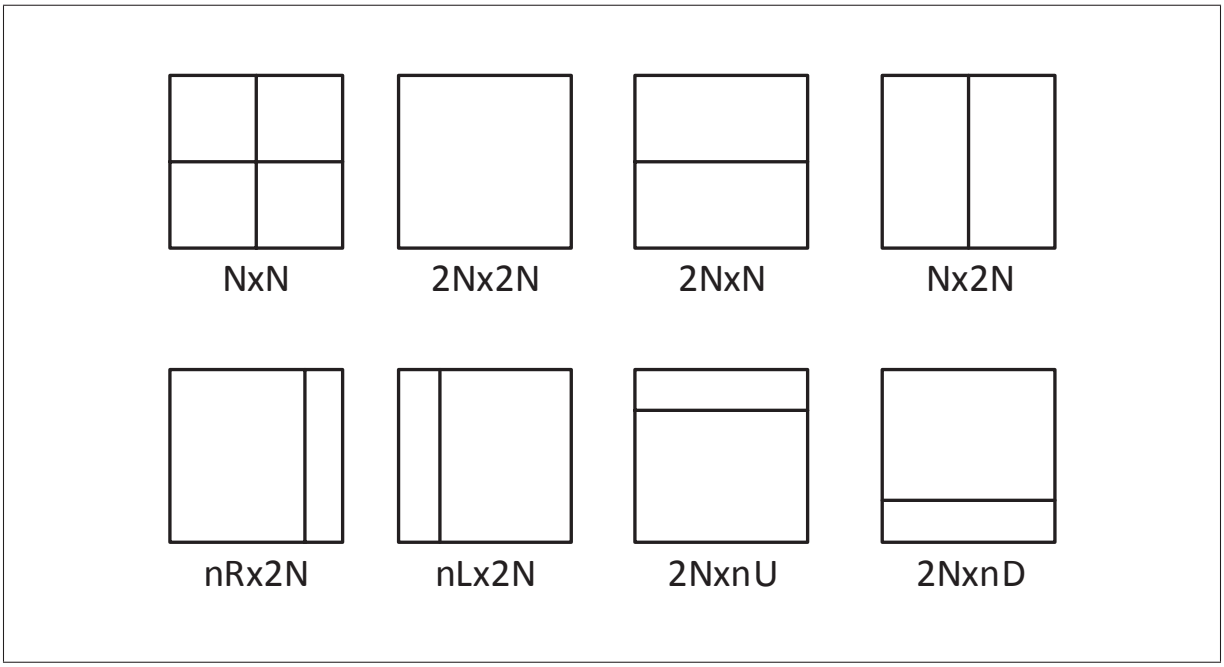

Figure 2.5 All possible modes of partitioning a CU into PUs in HEVC. Intra CUs can apply only the first two modes.

As it could be seen, there are many possibilities to split a picture into multiple units and blocks and there are also many ways to combine the coding tools. Although this may not have a significant impact on the decoder complexity, the encoder should perform heavy computations

to leverage the full capabilities of the standard. This is mostly because the encoder should choose the best coding tree structure and the best way for the subdivision of a CU into PUs and TUs. This process can be performed by an exhaustive execution of RDO and is extremely time consuming. The RDO process considers all the encoding possibilities and compares them with regard to bit rate and picture quality. Since CU is the root for PU partitioning and the RQT configuration, it could be generally deduced that the computational complexity of RDO increases monotonically with the depth of the CU splitting (Ma *et al.*, 2013). Limiting the CU depth can reduce the complexity but decreases the coding efficiency because small CUs can cope efficiently with the regions of the picture with complex texture while large CUs cannot successfully cover these areas. The RDO problem is a very important one in complexity analysis of HEVC and determining the best partitioning pattern of CTUs in a picture in a reasonable time can dramatically decrease the complexity burden of the HEVC encoder.

2.3 HEVC Computational Complexity Analysis

In this section, we focus on the HEVC encoder from the complexity point of view and provide some analyses on the most complex parts of the encoder; the parts that need more attention for complexity reduction. Since the commonly used software for research and codec development is the HM, this implementation is considered as a test bed for evaluating the standard and three encoder configurations are used which are known as all-intra (AI), random access (RA) and low delay (LD). In some studies LD is considered as two different configurations called LP (or LD_P) and LB (LD_B). Considering the coding time as a measure of complexity, the first step for analyzing the encoder complexity is calculating the overall coding time for different video sequences and different encoder configurations. We have selected some papers that describe our problem in the best manner and help understand which parts of the codec are most complex and give guidelines to reduce this complexity.

A complexity and implementation analysis of the HEVC encoder and decoder is presented in (Bossen *et al.*, 2012). For complexity analysis, first they consider the overall time that the encoder and the decoder consume for processing different sequences to show that real-time

video coding is not easy to achieve with current implementation of HEVC. Tables 2.1 and 2.2 from this paper present some common sequences and the encoding time of HM for encoding these sequences. All of the test sequences are 10 seconds long and two values of 27 and 32 are considered for quantization parameter (QP); AI27 means all-intra with the quantization parameter (QP) equals to 27 and so on. These results are obtained using gcc 4.4.5 for compiling and a cluster with Xeon-based servers (E5670) clocked at 2.93 GHz.

Table 2.1 Test sequences, (Bossen *et al.*, 2012)

Sequence	**Resolution**	**Frame Rate (Hz)**
Kimono	1920×1080	24
ParkScene		24
Cactus		50
BasketballDrive		50
BQTerrace		60
BasketballDrill	832×480	50
BQMall		60
PartyScene		50
RaceHorses		30

Table 2.2 Encoding time for test sequences, times are in tens of seconds, (Bossen *et al.*, 2012)

	Time (10s)					
Sequence	**AI27**	**AI32**	**RA27**	**RA32**	**LB27**	**LB32**
Kimono	393	357	1283	1123	2016	1739
ParkScene	462	395	1145	1000	1743	1501
Cactus	955	811	2590	2257	3635	3133
BasketballDrive	870	759	3155	2707	4417	3793
BQTerrace	1228	1043	2936	2485	4029	3315
BasketballDrill	194	166	606	515	826	700
BQMall	229	202	642	562	900	779
PartyScene	245	210	614	505	882	724
RaceHorses	120	104	481	396	686	570

As can be seen from the tables even in only intra, the coding time can be more than 1000 times of the sequence length and this makes real-time coding very challenging. To overcome this problem, we should know which modules of the encoder contribute the most to the complexity. In this regard, (Bossen *et al.*, 2012) analyzes different C++ classes of the HM that relate to various parts of the codec. This analysis gives a sense on how much time is spent in each module of HEVC. Table 2.3 illustrates the classes and the time spent in each of them. This is shown for two AI and RA configurations. As it could be seen from the table, in AI setting, a significant amount of time is spent in TComTrQuant class. This class is responsible for rate distortion optimized quantization (RDOQ). TComPrediction and TComPattern classes, both of them related to intra-picture prediction, account for 16%. For the core entropy coder, TEncBinCABAC class is used which is comprised of TEncBinCABAC and TEncBinCABACCounter classes and small amount of time is spent in them (about 2%). However, more time is spent on scanning and context derivation that happened in TEncSbac. Another important class for intra coding is TEncSearch. This class consumes 11.8% of encoder time for AI configuration. This class is responsible for finding the best prediction signal for the current block.

Table 2.3 Consumed time in encoder classes, (Bossen *et al.*, 2012)

	Time (%)	
Function	**AI**	**RA**
TEncSearch	11.8	7.4
TComTrQuant	24.4	10.7
TComRdCost	9.8	38.8
TComInterpolationFilter	0.0	19.8
TComYUV	0.1	1.7
partialButterfly	8.7	4.0
TComDataCU	5.8	2.7
TEncSbac	8.4	3.5
TEncEntropy	1.2	0.6
TEncBinCABAC	2.2	0.9
TComPrediction	10.0	1.1
TComPattern	6.6	0.4
memcpy/memset	11.0	7.1

Another good analysis of video coding complexity for HEVC is done in (Vanne *et al.*, 2012). The main focus of this paper is comparison of the HM and the joint model (JM) as HEVC and H.264/AVC reference software respectively. It also presents an analysis of rate-distortion complexity problem of HEVC. This work shows that HM (Main Profile) achieves bit rate savings of 23%, 35%, 40% and 35% relative to JM (High Profile) in AI, RA, LB and LP configurations respectively. As it is expected this higher coding efficiency comes with a cost to the HEVC codec from the complexity point of view. The paper analyzes all of the configurations for software complexity comparison and presents the complexity percentage of each module at the encoder. Due to the large share of integer motion estimation (IME), fractional motion estimation (FME) and mode decision (MD), the paper focuses more on these stages and considers their internal functions. The high complexity of these modules is attributed to interpolation (IPOL), SATD and sum of absolute differences (SAD) functions in FME, FME/MD and IME respectively. Table 2.4 shows the average complexity contribution of these modules and other parts like intra prediction (IP), transform/ quantization/ inverse quantization/ inverse transform (T/Q/IQ/IT) and entropy coding (EC) over all the tested sequences. Table 2.5 shows most complex functions at the encoder and their complexity shares in AI, RA, LB and LP configurations. In AI setting only SATD is present for MD process and in the other configurations these three functions are responsible for the majority of the encoding complexity. As another result of the analysis, SAD share is about 65% of the IME complexity while around 95% of the FME/MD complexity is because of IPOL and SATD.

Table 2.4 Average share of encoding complexity, (Vanne *et al.*, 2012)

Encoding stage	**AI**	**RA**	**LB**	**LP**
IME	0%	16%	18%	17%
FME/MD	9%	55%	59%	49%
IP	24%	1%	1%	1%
T/Q/IQ/IT	41%	14%	11%	18%
EC	11%	4%	3%	5%
Misc.	15%	10%	8%	10%

Table 2.5 Most complex functions at the encoder and their complexity shares, (Vanne *et al.*, 2012)

Function	**AI**	**RA**	**LB**	**LP**
Interpolation	0%	37%	38%	31%
SATD computation	9%	16%	18%	15%
SAD computation	0%	10%	12%	11%
Sum	9%	63%	68%	57%

2.4 HEVC Intra Coding

HEVC intra coding follows the same structure as previous hybrid video codecs (Lainema *et al.*, 2012). It is mainly based on spatial prediction and transform coding. However, it carries some new features, such as an increased number of prediction modes and a new frame splitting approach. The encoder uses spatial correlation in one frame to reduce the amount of data for transmitting or storing the visual information. To this end, it chooses the best coding unit depth, best transform unit tree and best mode for each prediction unit. An intra mode determines the direction to predict a block. HEVC intra coding supports 35 prediction modes for the luma component. There are 33 directional modes, allowing an efficient prediction of different directional video contents, a dc mode for homogeneous regions, and a planar mode to predict the smooth surfaces. Using the dc and planar modes let HEVC intra predict areas of the image which do not follow an edge model. Figure 2.6 shows HEVC luma intra prediction modes. In the early versions of the standard the number of modes which could be used to predict a block was dependent on the size of PU and only a subset of modes could be selected. Now, however, for all sizes of PU, all 35 modes are examined. The decision for coding a block as intra is made at the CU level, but the intra mode is selected for each PU, and it is possible for PUs in the same CU to have different intra modes. After the intra mode is selected, the prediction is done for TUs inside the PU. This means that in determining the prediction signal, the spatially neighboring TUs are used. For a TU with size $N \times N$, there are $4N+1$ samples for prediction from the above, above-right, above-left, left and below-left TUs. Samples from the below-left TU are not always available, and can only be used when they have been processed

and decoded beforehand. As an example a prediction by mode 2 is shown in Figure 2.7 for a 4×4 block.

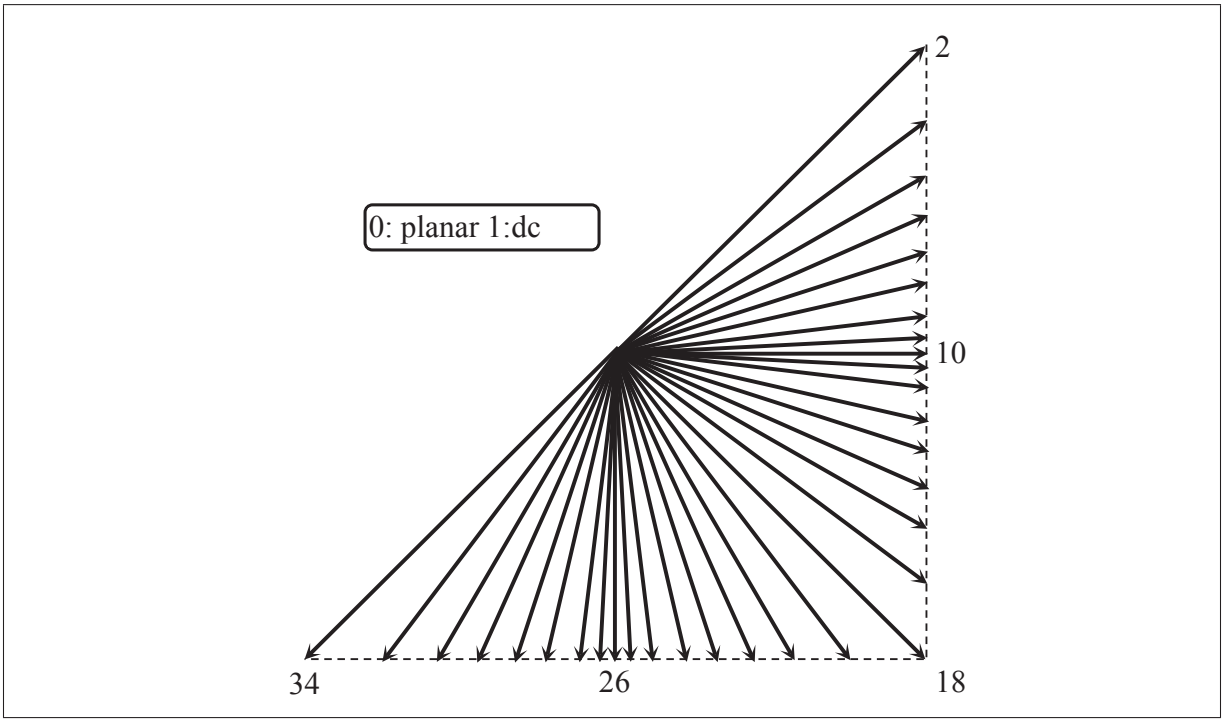

Figure 2.6 HEVC intra prediction modes.

Although this increased number of modes results in an improved coding efficiency, it renders the encoding process much more complex due to the increased number of RDO computations required. To lower this high complexity, the HM uses a two-step approach to find the best intra mode. The first step, rough mode decision (RMD), selects N candidates with the lowest RMD costs to be processed by the second step, RDO, where the candidate with the lowest RDO cost is selected as the best mode. Table 2.6 shows N for different PB sizes. Although this approach reduces the computational complexity to some extent, the HM encoder still needs to be improved substantially for real-time and fast encoding.

The RMD cost (C_{RMD}) is computed as:

$$C_{RMD} = SATD_l + \lambda_p \times R_p, \tag{2.1}$$

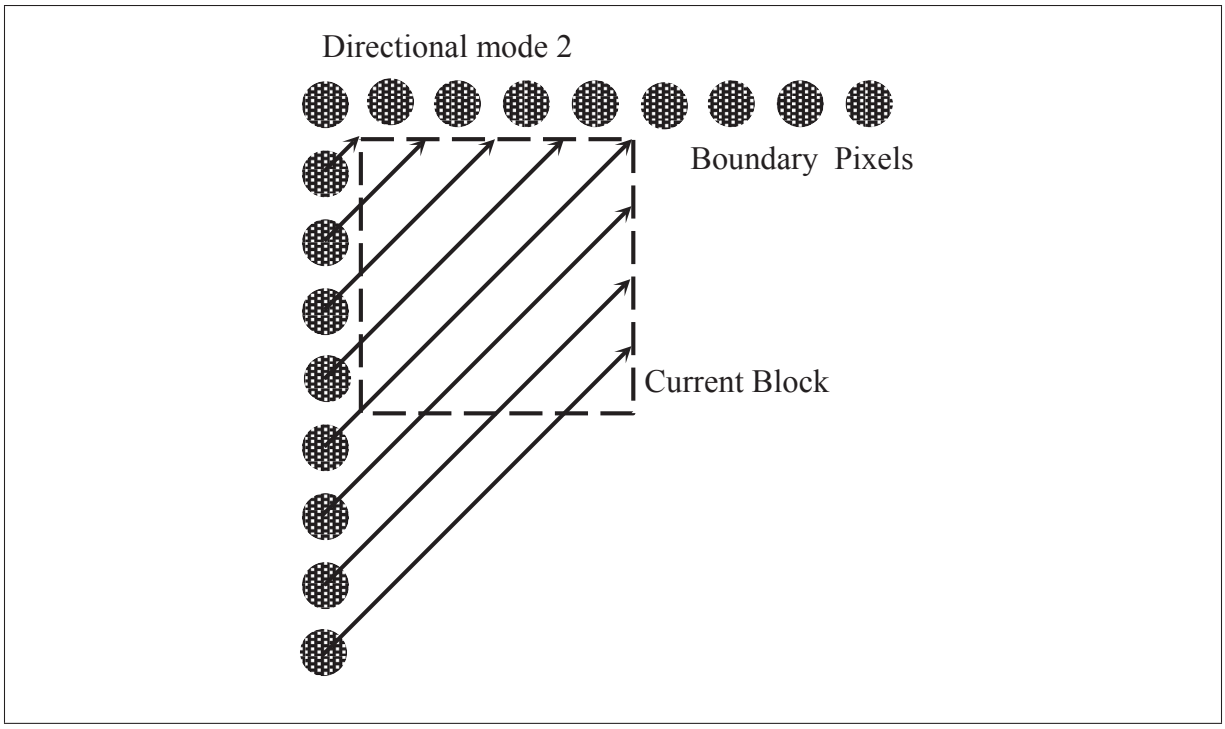

Figure 2.7 Intra prediction based on directional mode 2.

Table 2.6 Number of intra mode candidates, selected by the HM, for different PU sizes

PU size	64×64	32×32	16×16	8×8	4×4
N	3	3	3	8	8

where $SATD_l$, as a measure of distortion, is the SATD between the original luma component and its prediction block, R_p is the number of bits used for mode coding, and λ_p is the Lagrange multiplier, which is determined based on the QP (Sze *et al.*, 2014). The RDO cost (C_{RDO}) for the N selected candidates is computed using the following cost function:

$$C_{RDO} = (SSE_l + \omega_c \times SSE_c) + \lambda_m \times R_m, \tag{2.2}$$

where SSE_l and SSE_c are the sum of squared errors (SSE) between the original luma and chroma components and their reconstructed blocks, ω_c is the chroma weight (depending on the QP), R_m is the number of bits for PU encoding and λ_m is the Lagrange multiplier. It should be

mentioned that before RDO computations, a maximum of three most probable modes (MPMs) are added to the candidates list based on the best modes of the neighboring blocks. The final prediction mode is the candidate with the minimum RDO cost.

For chroma components, HEVC provides five intra modes, regardless of the block size, which include planar, dc, vertical, horizontal and the best luma mode. If the best luma mode is one of the first four modes, a diagonal mode (34) is used as the fifth mode. Using this concept of derived mode results in an efficient signaling when chroma content follows the same structure as the luma, and thus, the same intra mode, which is a very likely scenario. There is no rough mode decision for chroma intra prediction, and to select the best mode among the five candidates, the HM evaluates the RDO cost for all of them, which makes the chroma component a complex module as well. In section 3.1, we review some works which propose methods to reduce the encoding complexity by decreasing the number of modes to be processed by RDO.

CHAPTER 3

LITERATURE REVIEW ON HEVC INTRA CODING

In this chapter, we give an overview of the state-of-the-art works on complexity reduction for HEVC intra coding. In the last decade, many works have proposed different methods for reducing the complexity of HEVC and H.264 intra coding. These methods propose decreasing the number of modes processed by RDO (mode decision) (Ruiz *et al.*, 2016; Marzuki *et al.*, 2016; Pakdaman *et al.*, 2016; Tariq *et al.*, 2016; Wang and Siu, 2013; Park and Jeong, 2015; Gao *et al.*, 2015; Zhang *et al.*, 2015; Pan *et al.*, 2005; Li *et al.*, 2009), or avoiding exhaustive search to find the optimal CU sizes inside a CTU (fast CU size decision) (Cho and Kim, 2013; Saurty *et al.*, 2015; Shen *et al.*, 2014; Fang *et al.*, 2013; Min and Cheung, 2015), or a combination of these techniques (Zhang *et al.*, 2017a; Shang *et al.*, 2016; Zhang and Ma, 2014; Song *et al.*, 2017; Zhao *et al.*, 2014, 2012; Zhang and Ma, 2013; BenHajyoussef *et al.*, 2017; Liu *et al.*, 2016). In the following, we present the major works in these two categories separately. Since many works combine the two techniques, we will see some overlaps in the following sections.

3.1 Mode Decision Complexity Reduction

In this section, several works regarding mode decision optimization while focusing on intra mode decision are presented. In intra mode decision, the goal of the researches have been mainly reducing the number of modes that enter the RDO process.

For mode excluding, (Yan *et al.*, 2012) have proposed a group-based intra mode decision approach to accelerate the encoding process. The idea is based on a statistical analysis of the modes generated by RMD. As mentioned before, RMD is a process that is performed before rate distortion calculation to reduce the number of intra modes. It calculates a cost for each prediction unit by applying Hadamard transform on residual between the original prediction block and its prediction and then computes the SATD for the transform coefficients. In (Yan *et al.*, 2012), the output modes of the RMD are found to be adjacent together in most cases, and

as a result, represent same direction. In addition, they reduce the number of candidate modes for RDO by merging adjacent modes into some groups and also performing a pixel-based edge detection. Early termination of block partitioning is another tool that is used in this work to reduce the time consumed in the intra mode decision module. It has been shown that for blocks with the size of 16×16, 32×32 and 64×64, only one mode from the RMD process is selected. For blocks with the size of 4×4 and 8×8 selecting only one mode depends on the difference between the two lowest RMD costs among the 35 modes. It has also been shown that the more the adjacency of the modes generated by RMD is, the better the results become.

In (Ma *et al.*, 2013), a low-complexity rate distortion optimization method for complexity reduction has been proposed. This method is comprised of three stages: fast intra mode decision, adaptive reference selection and CU splitting early termination. It is noted that this method is applied to the most complex stages at the encoder. The method uses the directional information from the neighboring blocks and the correlation between energy of prediction residuals and CU splitting for intra and inter-prediction respectively to early terminate CU splitting. The energy of prediction residuals is defined as:

$$E = \frac{1}{m \times n} \sum_{i=0}^{m} \sum_{j=0}^{n} r_{i,j}^2. \tag{3.1}$$

In addition, the method employs the spatial and temporal correlations among the neighboring frames and CUs to limit the number of reference frames. It has been shown that by using these three techniques, it is possible to implement HEVC for complexity-constrained encoders.

Another work on intra mode decision by applying mode excluding has been presented in (Zhang *et al.*, 2012). This work claims that the RMD is not a perfect strategy for reducing the number of intra modes, since it is the same for different blocks and does not consider their variations. Accordingly, an adaptive approach for fast intra mode decision has been proposed. In this work, based on statistical analysis, it has been shown that it is highly likely that the first two candidates in RMD and especially the first one is selected by the RDO process as a final optimum mode. It has also been observed that in blocks with highly complicated texture or

blocks with smooth content, dc and planar modes have more chances of being selected as the lowest cost mode.

There are some works on intra mode decision that have used texture characteristics to decide which modes are the most probable ones to win the RDO race. Most of them have employed edge detection algorithms based on gradient to determine the best matched modes for a specific PU. In (Chen *et al.*, 2013), a fast mode decision algorithm to lighten the computation complexity in intra mode prediction has been proposed. The method is based on edge detection and includes a sub-method for depth decision. The work takes the most probable modes from the neighboring blocks to improve the accuracy of the intra mode prediction.

A method has been proposed in (da Silva *et al.*, 2012) to use edge information of the current prediction unit in addition to the modes of the neighboring PUs to reduce the number of modes that are going to be tested by the rate distortion optimization process. The idea is based on categorizing the edges into five groups namely horizontal, vertical, 45 degree, 135 degree and one non-directional edge. The algorithm uses the pixel values to select one of these five directions as a dominant edge for the prediction unit, and thus, limited modes are selected for cost calculation which reduces the encoding time.

Another work on intra mode decision using gradient and edge concepts has been presented in (Jiang *et al.*, 2012). In this work, a gradient-based fast mode decision algorithm has been proposed to reduce the computational complexity of the encoder. The gradient directions are determined before the intra prediction, and for each PB, a gradient-mode histogram has been obtained. Consequently, limited numbers of modes are selected for the final RMD and RDO processes.

(Wang *et al.*, 2014) have presented a fast intra coding algorithm. In this work, the number of modes in RMD and RDO processes is reduced by taking into account the correlation among neighboring blocks. A depth range prediction method has also been proposed by exploiting the correlation between the neighboring CTUs. To this end, they define three prediction depth

ranges:

$$R0 = [0,3],\ \ R1 = [0,2],\ \ R2 = [1,3] \tag{3.2}$$

The depth range $R1$ is used for homogeneous regions and the depth range $R2$ is used for inhomogeneous regions. Depth range $R0$ would be used if it is not possible to determine the homogeneity of the block. The depth range of the current CTU, DR_{curr}, is defined as:

$$DR_{curr} = \begin{cases} R1, & D_{max}^{left} \leq 1 \ \& \ D_{max}^{up} \leq 1 \\ R2, & D_{max}^{left} > 1 \ \& \ D_{max}^{up} > 1 \\ R0, & other \end{cases} \tag{3.3}$$

where D_{max}^{left} and D_{max}^{up} are the maximum depths of the left and upper CTUs, respectively.

In (Ruiz *et al.*, 2016), a novel algorithm has been presented based on the orientation detection, which uses the local directional variance along some predefined lines. Based on this analysis, orientations with the lowest directional variances are considered as dominant orientations, and the number of intra mode candidates to be further processed by RDO is reduced accordingly.

The authors in (Marzuki *et al.*, 2016) have proposed a context-adaptive fast intra coding based on the best modes of the upper CU and neighboring PUs. The method includes early termination approaches for RDO as well as RQT using adaptive thresholding.

In (Pakdaman *et al.*, 2016), the authors have proposed a novel method based on the dual-tree complex wavelet transform (CWT) to effectively determine edges leading to the selection of the most appropriate candidates for intra mode decision. This approach considers only few adjacent candidates for the best estimated mode selected by the edge analysis. It is noted that CWT decomposes an image into various frequency sub-bands. The algorithm uses the energy distribution in these sub-bands to exploit texture information and obtain the dominant direction for each block.

The authors in (Tariq *et al.*, 2016) have proposed a method for intra coding complexity reduction based on the following quadratic relation between the RDO cost and SAD:

$$RD_{\text{cost}} = \alpha_1 SAD^2 + \alpha_2 SAD + \alpha_3. \tag{3.4}$$

Using this method, it is possible to avoid entropy coding, Hadamard transform, and distortion computations for mode selection and thus save time and computations. The method also formulates the distortion and rate as functions of SAD and QP, in order to avoid performing the RDO process.

A method has been presented in (Wang and Siu, 2013) for fast intra HEVC coding based on three optimized candidate sets including 1, 19 and 35 modes. Using the neighboring reference samples, the encoder selects the optimal set for each PU. This results in an accelerated encoder due to the reduced number of modes needed to be processed by RDO for the first two sets. Further, the number of bits needed for mode signaling is reduced.

In (Park and Jeong, 2015), the PU has been converted from the pixel domain to the transform domain, and the main directions have been determined in the latter. Based on these directions, a short list of candidate modes is selected for the RDO process.

The authors in (Zhang *et al.*, 2017a) have proposed a low-complexity HEVC intra coding method based on both fast mode decision and early CU size determination. For the fast mode decision, a gradient-based approach is used to reduce the number of candidate modes to be processed by RMD and RDO. For the fast CU size decision, texture homogeneity along with a support vector machine (SVM)-based approach are used to make early CU splitting and early CU termination decisions. The SVM uses the depth differences, Hadamard cost and rate-distortion (RD) cost as features.

In (Shang *et al.*, 2016), a low-complexity intra HEVC encoder has been proposed based on a fast PU mode decision (FPUMD) and a fast coding unit size decision (FCUSD). The FPUMD reduces the number of intra mode candidates based on the correlation of the PU mode and RD

cost of the different depth levels. The FCUSD for its part excludes unnecessary CU sizes from further processing, using the depth of neighboring CUs and a RD cost threshold determined from previous frames. To update the coding parameters, an online method is used leading to an accurate decision for various sequences types.

In (Zhang and Ma, 2014), a method has been proposed for fast intra coding, and achieves significant time reduction by using the Hadamard cost-based progressive rough mode search (pRMS) and early CU splitting termination. Using pRMS, the algorithm selectively checks the potential modes instead of traversing all candidates. This allows fewer modes to enter the RDO process. Further, the early CU split termination excludes the lower depths if the estimated RD cost (aggregated cost of the sub-CUs) is already larger than the RD cost of the current CU.

In (BenHajyoussef *et al.*, 2017), a gradient-based pre-processing step has been proposed to reduce the number of intra mode candidates. A gradient-based approach has also been developed to make an early decision for CU splitting. This approach is based on texture complexity and CU sizes.

In short, several works have proposed notable and novel ideas regarding intra mode decision complexity reduction. However, they exhibit some shortcomings, which could be addressed in a bid to design even faster HEVC intra encoders. In gradient-based approaches, gradients could be computed at the CTU level, instead of at the PU level, and then reused for every block inside the CTU. This makes it possible to use the same edge information at each depth and to avoid repetitive calculations for each block. In addition, since the picture edges are not perfectly aligned with intra modes, works in this area need to come up with a solid strategy to relate the gradients at each pixel to directional intra modes. Relying only on the modes of neighboring blocks as is the case in some works, may result in a domino effect, where a wrong decision can propagate to other blocks and affect coding efficiency negatively. To avoid this drawback, these kinds of algorithms need to be accompanied by other methods. Other works aimed at reducing the number of modes for RDO process based on low-complexity measures keep N best modes as a candidates list. Using a fixed number of candidates, they either waste

computations by including modes with no potential to be the best mode or reduce the coding efficiency by excluding the best mode. In addition, the methods which select an adaptive number of candidates end up with limited time reduction or high quality loss due to imperfect RDO candidates selection. In chapter 4, we have proposed our methods for intra mode decision complexity reduction to improve other approaches in this area by addressing the shortcomings noted.

3.2 Coding Unit Size Decision Complexity Reduction

It is known that HEVC introduces a new kind of picture partitioning in hybrid video coding; the picture is partitioned into CUs, PUs and TUs in a very flexible manner to improve the coding efficiency as much as possible. The structure of these three basic processing blocks let the encoder do the encoding process optimally and improve the compression capability significantly. Even though this highly flexible scheme has been successful in increasing the coding performance, it imposes a huge amount of computations especially on the encoder. These computations mostly come from the exhaustive rate distortion cost calculation for all the combinations of the units, forcing the video coding community to find solutions to reduce this complexity and alleviate the codec structure. Generally, in this area, researchers look for methods which exclude some combinations of these blocks and try to avoid examining all depths of the coding unit by early termination of the splitting process or early splitting of the coding unit. In this section, a review of some of these works are presented.

In (Yu *et al.*, 2012), a method for early termination of CU splitting has been proposed to reduce the complexity. Although the method has been developed for inter coding, it may also be used for intra coding. The method is based on the fact that for temporally stationary and spatially homogeneous regions, the residuals are small and the use of large CUs is the efficient way of partitioning. In addition, for early terminating the CU splitting, the characteristics of prediction residuals have been used. This approach is based on computing the mean squared errors (MSEs) between the prediction and the original block for the current CU level and comparing it with an adaptive threshold. This comparison forces the CU splitting process to

early terminate. It is noted that the threshold is set adaptively for each depth and an adjustable factor tunes its impact on complexity reduction and coding efficiency. Although this method is useful for homogeneous and stationary regions, it may lead to more computations in picture areas with high spatially and temporally changes due to considering an additional threshold parameter.

A method for fast CU size decision, using machine learning, has been presented in (Liu *et al.*, 2017). In this work, CU complexity classification (CC) is used to adaptively determine the size of the current CU. To obtain the CC, the most important features related to the size of the CU are extracted. Having the features at hand, the classification model of the CU is constructed using SVM. The most important features they use are the block complexity, directional complexity and difference complexity. The block complexity is represented by variance as follows:

$$\begin{aligned} var &= \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(h(i,j)-\bar{h})^2 \\ \bar{h} &= \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}h(i,j), \end{aligned} \tag{3.5}$$

where N is the block size and $h(i,j)$ and $\bar{h}$ are the pixel value and the mean pixel value of the block, respectively. The directional complexity is represented by:

$$DCom = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(G_{hor}(i,j)+G_{ver}(i,j)+G_{45}(i,j)+G_{135}(i,j)), \tag{3.6}$$

where G is the gradient computed in different directions. Finally, the sub-CU complexity difference is defined as:

$$SCCD = \frac{1}{4}\sum_{i=0}^{3}(var_i-\overline{var})^2, \tag{3.7}$$

where var_i is the variance of a sub-CU and $\overline{var}$ is the average variance of four sub-CUs.

(Shen *et al.*, 2014) have proposed a method for fast CU size decision which speeds up the splitting process by reducing the number of candidate CU sizes. This method employs the texture homogeneity to achieve adaptive thresholds for early determination of the CU size.

In addition, a combination of texture property and sizes of the neighboring CUs are used to develop a bypass strategy for CU size decision. To determine the block homogeneity, they use the concept of mean absolute deviation (MAD) for the overall block and horizontal and vertical directions as follows:

$$MAD = \frac{1}{u \times v} \sum_{y=0}^{u-1} \sum_{x=0}^{v-1} |p(x,y) - m| \tag{3.8}$$

$$MAD_H = \frac{1}{u \times v} \sum_{y=0}^{u-1} \sum_{x=0}^{v-1} |p(x,y) - m_y| \tag{3.9}$$

$$MAD_V = \frac{1}{u \times v} \sum_{y=0}^{u-1} \sum_{x=0}^{v-1} |p(x,y) - m_x|, \tag{3.10}$$

where $p(x,y)$ is the pixel value and m is the average pixel value of the CU. In addition, m_y is the mean of pixel values at the yth row and m_x is the mean of pixel values at the xth column. Then, texture homogeneity of the CU is determined as follows:

$$\begin{cases} homogeneous, & if \quad min(MAD_H, MAD_V, MAD) \leq T \\ complex, & otherwise \end{cases} \tag{3.11}$$

where T is a threshold.

In (Lee and Jeong, 2017), a method has been proposed for fast CU size decision using statistical information. This method uses the image complexity along with an adaptive depth prediction process to early determine the size of the CU. The CU splitting is terminated early based on the results of a Bayesian classification and a quadratic discrimination analysis.

In (Zhang *et al.*, 2013), an approach for accelerating the CU depth decision has been proposed to reduce the complexity of the encoding process. This approach is based on limiting the search range by using the spatial and temporal correlation between the current block and adjacent blocks as well as excluding some depths from the CU splitting tree.

(Kim *et al.*, 2012) have proposed a method for early termination of CU splitting tree. This method is based on the rate-distortion analysis of the prior skipped CUs and adjusts the

weighting factor of a previously proposed scheme FEN (fast encoder decision algorithm). If the probability that the current CU is coded in the skip mode is very high, the weighting factor may be increased. In addition, the method assumes that the current CU mode is affected by neighboring and higher depth CUs. To demonstrate the effectiveness of the method for complexity reduction, experiments are conducted for random access configuration and high efficiency mode.

In order to reduce the complexity of CU splitting, a method has been proposed in (Shen *et al.*, 2012), which is based on Bayesian decision rule. The first step in this scheme is to divide the current CUs into the two classes according to the decision of splitting or not splitting them. In this classification problem, the current CU belongs to W_N or W_S classes if it is not split or split into four equal sub-CUs, respectively. Taking into account the posteriori probabilities of the classes for each CU and using the Bayesian rule, a structured classification problem has been presented.

In (Zhang and Ma, 2014), a method for fast CU size decision has been proposed using the Hadamard cost-based progressive rough mode search (pRMS) and early CU split termination. By using pRMS, the algorithm selectively checks the potential modes instead of traversing all candidates. This allows fewer modes to enter the high demanding rate distortion process. The early CU split termination is considered at macro-level and excludes the lower depths if the estimated rate distortion cost (aggregated costs of the sub-CUs) is already larger than the rate distortion cost of the current CU.

A method on fast CU size decision has been proposed in (Ruiz-Coll *et al.*, 2014), which is a fast partitioning algorithm using machine learning. To this end, a new approach for accelerating the CU partitioning decision has been developed based on trained data mining trees. In other words, a decision tree using a low complexity attributes such as mean and variance of blocks is constructed, which allows an early classification of the CU and avoids the RDO calculations for the all sizes. As a result, the computations for selecting the best mode are run just for a few CU sizes.

In (Helle *et al.*, 2017), a CU size decision method for inter coding based on reinforcement learning (RL) is proposed. In this method, the encoding procedure is considered as a decision process in time which can be well modeled by reinforcement learning. The method is based on offline RL where the strategy is learned based on some training data and then it is applied to the encoding process. (Kim and Ro, 2018) is yet another method for CU size decision using machine learning. The method uses neural networks to predict the CU size and thus reduce the computational complexity. In this method convolutional, pooling and fully connected layers are used to well analyze the CU properties, and based on these properties, the network is able to determine the size decision of the CU.

To summarize, it should be noted that the most successful algorithms are the ones predicting the homogeneity of the block with more accuracy. This helps the algorithm to decide whether the block should be split or not. The criteria that are used for this purpose include variance, mean, energy of the residuals and neighboring CUs' depth. The more accurate in finding this homogeneity, the more successful the algorithm is. Similar to mode decision, the use of the neighboring blocks for determining the current block's depth has been considered in some papers. The spatial and temporal neighboring blocks in addition to the blocks from other depths are employed. In addition, the rate distortion costs of the current depth and previous depths have been used to decide whether or not the block should be further split. It is noted that in chapter 5, we have proposed new methods for CU size decision complexity reduction to improve the state-of-the-art in this area.

CHAPTER 4

PROPOSED INTRA MODE DECISION METHODS

In this chapter, we introduce our proposed methods for complexity reduction in HEVC intra coding mode decision. These methods intend to discard irrelevant modes from further processing, adding highly likely neighboring modes, dodging RDO process and predicting RDO cost based on the low-complexity RMD cost. We start by RDO cost prediction as presented in the following section.

4.1 Mode Decision Based on RDO Cost Prediction

To reduce intra mode decision complexity, we add new modules to the encoder. Figure 4.1 shows the luma mode decision of the proposed method compared to the standard implementation of the encoder at the PU level, where it illustrates the mode decision steps. In this diagram, N_{HM} and N_L are same as N in Table 2.6. The RMD process is replaced by a gradient analysis step to select N_L luma promising candidates. Since we apply gradient computations for each CTU once and use the same information for all PUs inside the CTU, the process is much faster than the original RMD. The gradient-based algorithm is explained in section 4.3. Then, to exploit the spatial correlation, the same approach as HM is used to add three MPMs from neighboring blocks. In the next step, a statistical model is proposed allowing the RDO cost to be predicted without the need to perform the RDO process. Next, based on this model, a reduced and adaptive number of candidates are selected to be processed by the high-complexity and high-demanding RDO. The entire algorithm keeps K_L promising candidates, which based on their RDO cost prediction, have a high potential to be the best intra luma mode for the current PU.

4.1.1 RDO Cost Modeling

A predictive model for RDO cost (C_{RDO}, Eq. (2.2)) is developed in this section based on a low-complexity measure to avoid the RDO process for non-promising candidates. Examples of

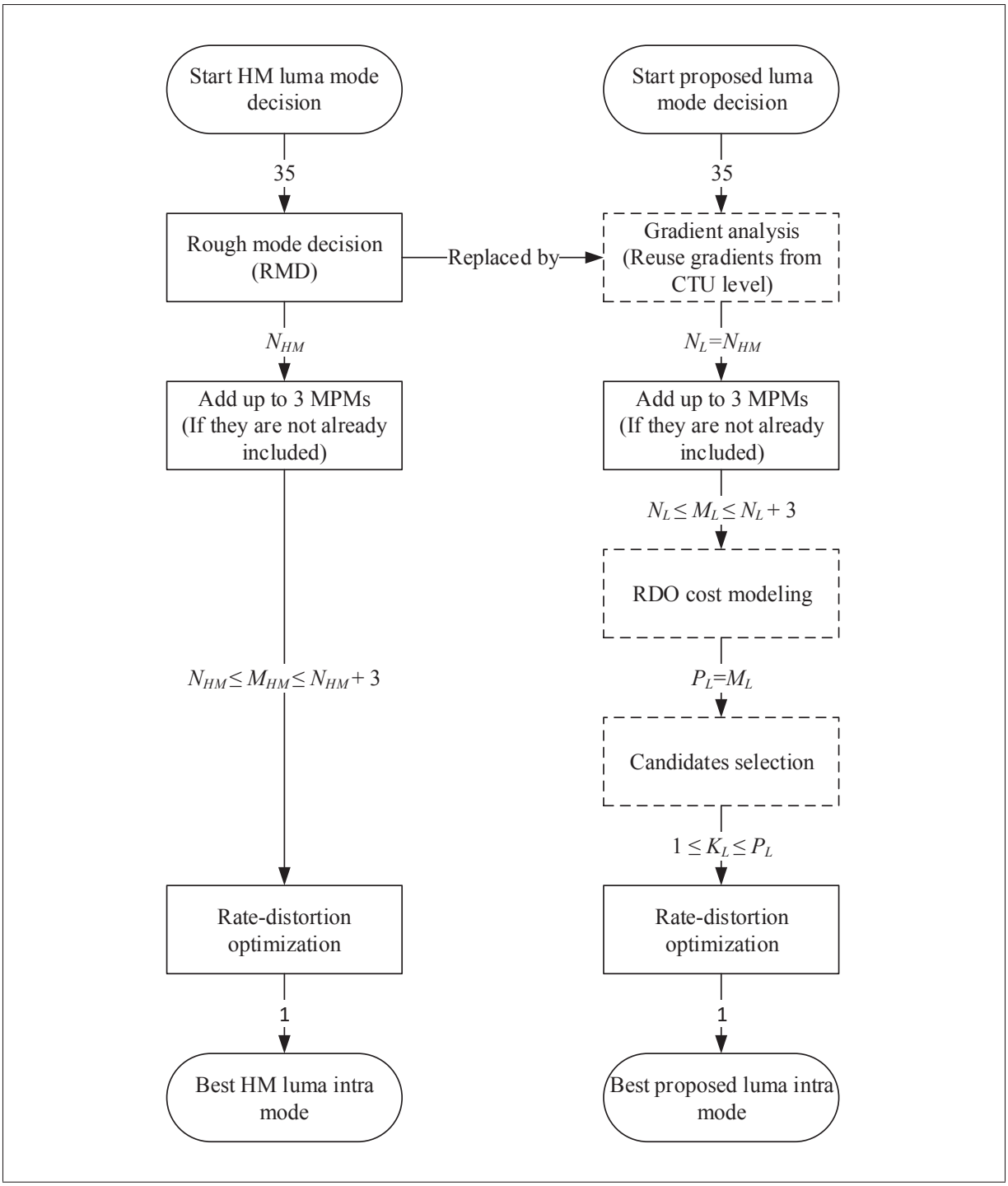

Figure 4.1 Block diagram of luma mode decision at PU level for HM (left) and proposed method (right). Arrows show the number of modes in each step.

such low-complexity measures include, but are not limited to, the RMD cost (C_{RMD}, Eq. (2.1)) which we use in this section. Figure 4.2 shows the correlation between the RMD cost and the RDO cost for a sample sequence, block size and QP. For this example, the Pearson correlation coefficient and the Spearman's rank correlation coefficient are 0.90 and 0.96, respectively. Similar results are observed for other sequences, block sizes and QPs. The Spearman's coefficient is more relevant to our RDO cost prediction goal since we are not looking for a linear correlation between two variables (as Pearson coefficient does), but rather, for rank correlation. That means if the relationship between the RMD cost and the RDO cost could be described by a monotonic function (Spearman's coefficient = 1), then the mode with the lowest RMD cost would be the same one with the lowest RDO cost, and in that case, the entire RDO process could be omitted.

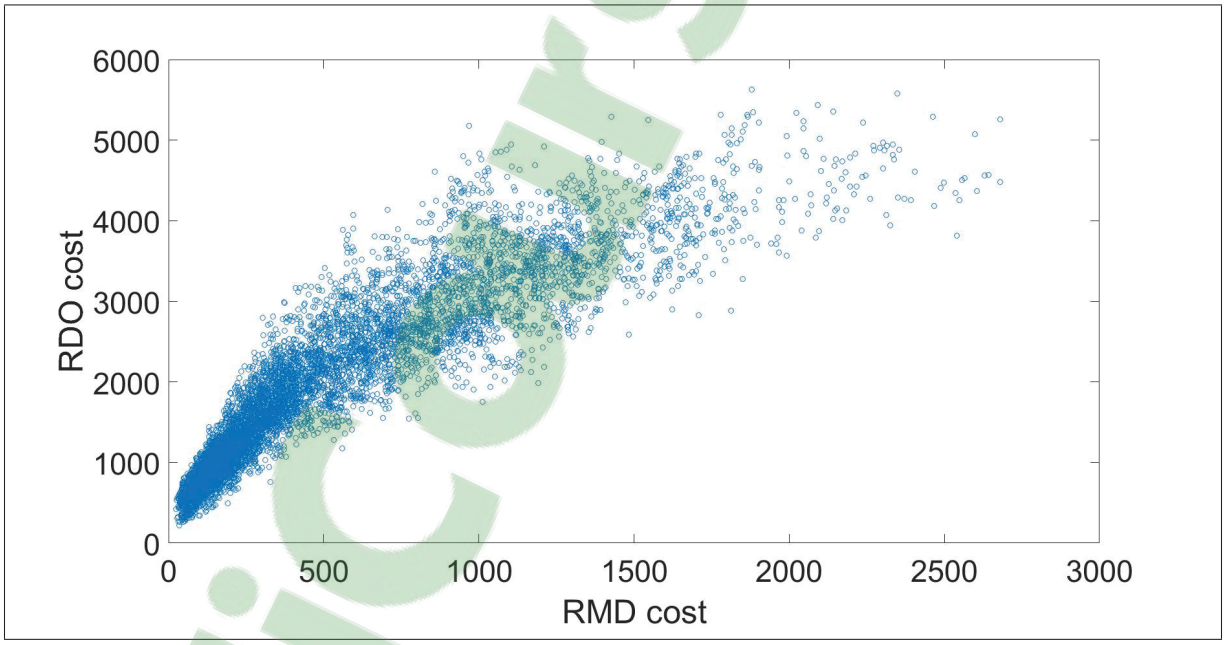

Figure 4.2 Scatter plot for RMD cost and RDO cost, *RaceHorses*, 4×4 blocks, QP=32.

Since there is no monotonic function describing the relationship between the RMD cost and the RDO cost, we propose a statistical model allowing the possible values of the RDO cost to be predicted from the RMD cost. Based on our observations, for a PU with a given C_{RMD},

or more specifically, with a given small range of C_{RMD} values, the C_{RDO} can be modeled by some well-known probability distributions. We compare some distributions based on the Bayesian information criterion (Bhat and Kumar, 2010) (BIC) to determine which of them can best describe empirical data. They are: Beta, Birnbaum-Saunders, Exponential, Extreme value, Gamma, Generalized extreme value, Generalized Pareto, Inverse Gaussian, Logistic, Log-logistic, Lognormal, Nakagami, Normal, Rayleigh, Rician, t location-scale and Weibull.

As mentioned above, although we cannot predict a specific value for C_{RDO} based on C_{RMD}, we can predict the probability distribution function (PDF) of its values, i.e., the likelihood of each RDO cost value. Figure 4.3 shows the top four models best fitted to the empirical C_{RDO} for a small range of C_{RMD} values for the *RaceHorses* sequence. We observe similar results for other video sequences with different QP values and block sizes with varying averages and variances. Having these models for all PU candidate modes allows us to omit non-promising candidates based on their C_{RDO} distribution, and to select an adaptive number of modes to go through the RDO process. In most cases, the best fitted model is normal, and we provide numeric results to support this claim.

Goodness of fit test

Although the data histograms are bell-shaped and similar to the normal probability curves, we need a more formal method to show that the RDO costs come from normal distributions. To this end, we need to perform a normality test. This allows us to determine if the costs are well-modeled by a normal distribution. The normality test could be performed using the descriptive statistics, hypothesis testing or Bayesian statistics approaches (Yazici and Yolacan, 2007). In descriptive statistics, a goodness of fit of a model is measured to determine how well it describes the data and how the observed values are different from those predicted by the model. To this end, tests such as Kolmogorov-Smirnov (KS), Anderson-Darling (AD) and Chi-Square (CS) can be used. Among these, the latter is frequently referred to in the literature as an accurate test that can effectively determine whether the expected data and observed data are significantly different from one another (Yazici and Yolacan, 2007). The Chi-Square goodness-

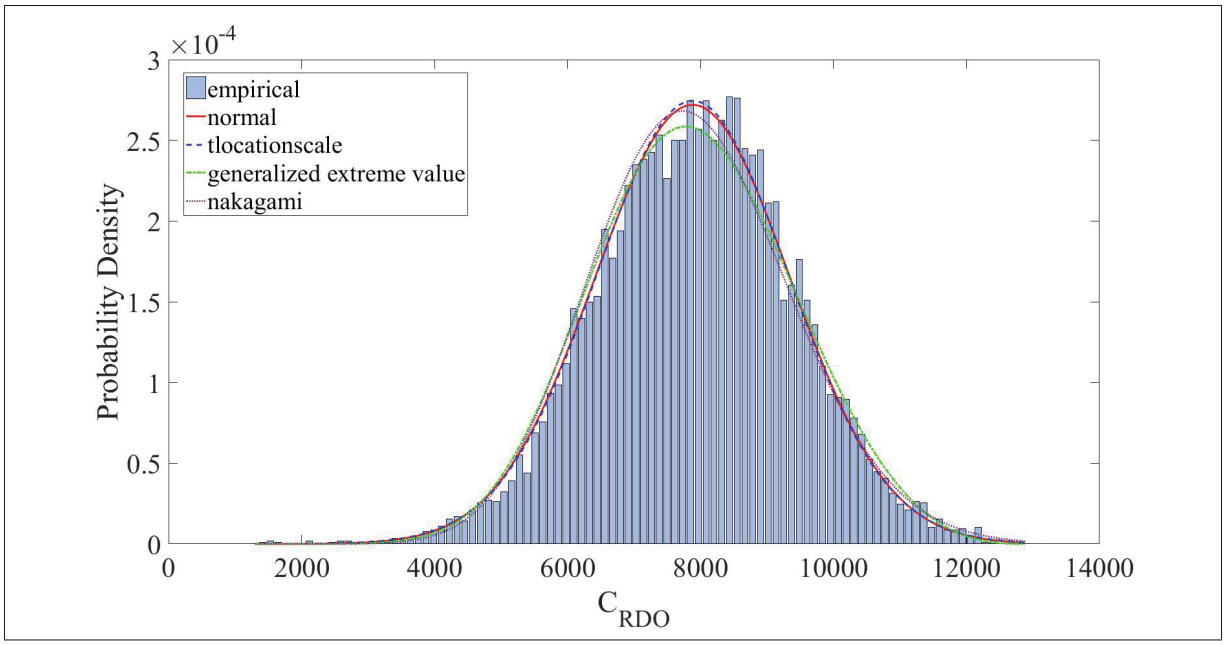

Figure 4.3 Best fitted distributions for C_{RDO}, *RaceHorses*, 8×8 blocks, QP=32, 2903<C_{RMD}<2945.

of-fit test uses parameters estimated from the data to validate that they come from a specified probability distribution. It puts the data into different bins, and based on the observed and expected population of these bins, it computes the Chi-Square test statistic as follows:

$$\chi^2 = \sum_{i=1}^{N} (O_i - E_i)^2 / E_i, \tag{4.1}$$

where O_i and E_i are the observed and expected counts for each bin, respectively. This test statistic is a Chi-Square random variable. Table 4.1 shows sample results for the mentioned tests for the normal model. In this table, the p-value is the probability of finding the observed, or more extreme, values when the null hypothesis (i.e., data comes from a normal distribution) is true. If the p-value is greater than a significance level, the test decision or h-value is zero, otherwise it is 1. A recommended significance level is 0.05, which works well (Yazici and Yolacan, 2007). The test decision in our case is zero, which means that the observed data is not statistically different from what is expected from a normal distribution.

The same interpretation could be made for Kolmogorov-Smirnov and Anderson-Darling tests. Similar results are observed for different sequences with various block sizes and QPs.

Table 4.1 Goodness of fit test results for the normal model, *RaceHorses*, 8×8 blocks, QP=32

	CS	**AD**	**KS**
Significance level	0.05	0.05	0.05
h-value	0	0	0
p-value	0.49	0.07	0.33

4.1.2 Candidates Selection

From the previous sub-section, we consider a normal model for the RDO cost of each candidate mode. Figure 4.4 shows these models for a sample PU, where $P_L = 8$. The RDO cost of each candidate mode $m_i\,(1 \leqslant i \leqslant P_L)$, with RMD cost of C_{RMD_i}, is represented by a random variable

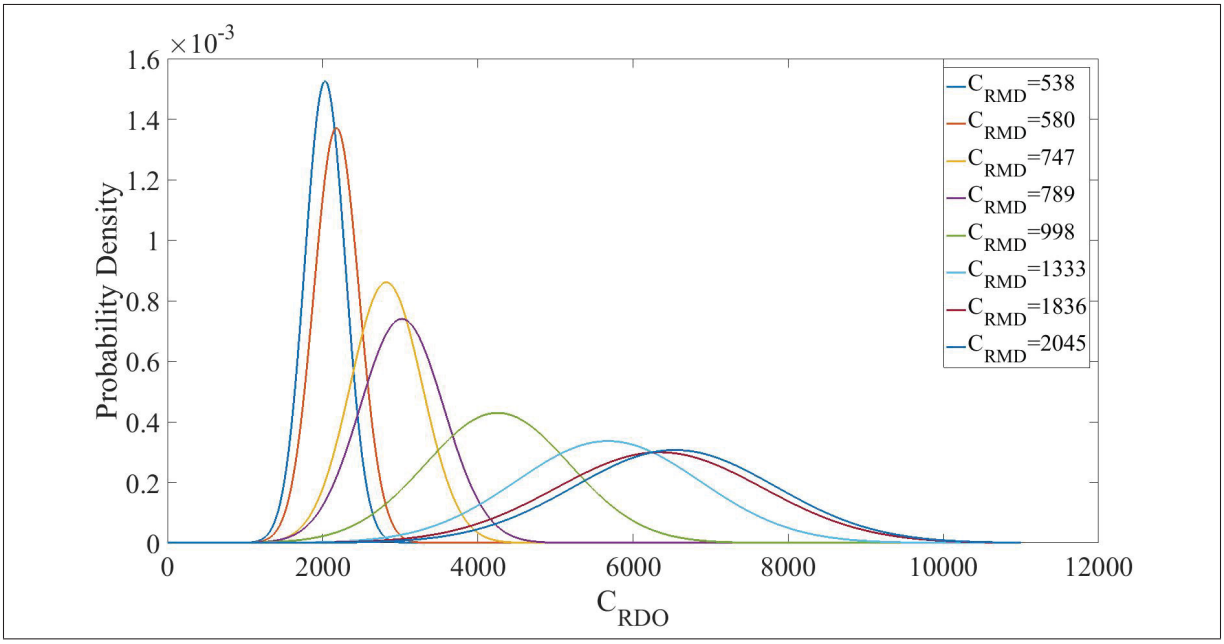

Figure 4.4 Normal distributions for C_{RDO} from a sample PU of size 8×8, *RaceHorses*.

X_i which follows a normal distribution with the following parameters:

$$\begin{aligned} \mu_i &= f_\mu(C_{RMD_i}) \\ \sigma_i &= f_\sigma(C_{RMD_i}), \end{aligned} \tag{4.2}$$

where μ_i and σ_i are piecewise constant functions of RMD cost and obtained using histograms similar to the one depicted in fig. 4.3. In the coding phase for any candidate mode with a given RMD cost, these parameters are obtained based on the available functions and so the RDO cost distribution for the candidate mode is known. By using these distributions, we exclude those candidates with a low chance of being the best PU mode. To select the K_L promising candidates, we consider the probability that a particular candidate is the one with the lowest RDO cost. The probability that mode m_i is the best RDO mode is $P(X_i < Z_i)$, where

$$Z_i = \min_{\substack{1 \leqslant j \leqslant P_L \\ j \neq i}} X_j. \tag{4.3}$$

Z_i is a random variable representing the minimum of the other $P_L - 1$ random variables. Then the following inequality is checked for all P_L candidates:

$$P(X_i < Z_i) > CL, \quad i = 1 \ldots P_L, \tag{4.4}$$

where CL is a confidence level. Those candidates which satisfy the condition are selected as promising candidates, and the RDO process is run only for them. The above probability can be written as follows:

$$P(X_i < Z_i) = \int_{-\infty}^{+\infty} P(X_i = t) P(Z_i > t) dt, \tag{4.5}$$

where:

$$P(Z_i > t) = P(\wedge_{\substack{j=1 \\ j \neq i}}^{P_L} X_j > t), \tag{4.6}$$

where $\wedge$ is the 'logical and' symbol. Assuming the normal distributions are independent, we have:

$$\begin{aligned} P(Z_i > t) &= \prod_{\substack{j=1 \\ j \neq i}}^{P_L} P(X_j > t) \\ &= \prod_{\substack{j=1 \\ j \neq i}}^{P_L} \left(\frac{1}{2} \operatorname{erfc}\left(\frac{t - \mu_j}{\sigma_j \sqrt{2}} \right) \right). \end{aligned} \tag{4.7}$$

The last equality is based on the complementary cumulative distribution functions for normal distributions, where erfc is the following complementary error function:

$$\begin{aligned} \operatorname{erfc}(t) &= 1 - \operatorname{erf}(t) \\ &= \frac{2}{\sqrt{\pi}} \int_t^{\infty} e^{-s^2} ds. \end{aligned} \tag{4.8}$$

This gives us a mathematical formula that can be further expanded to have an analytical answer to the problem. However, implementing such a case, even with the independency assumption, is almost impractical.

To have a more practical and implementable solution, we break the problem into sub-problems consisting of comparisons of two normal distributions. In this approach, the best RMD mode (lowest C_{RMD}) and any other mode which is likely to be the best RDO mode (lowest C_{RDO}) constitute the set of K_L promising candidates. Considering the P_L random variables $X_i \sim N(\mu_i, \sigma_i^2)$ associated with P_L candidates, mode m_i is selected for the RDO process if:

$$P(X_i < X_1) > CL, \tag{4.9}$$

where, without loss of generality, we assume the μ_is are sorted in increasing order ($\mu_i \leqslant \mu_j, \ \forall\, i < j$) and m_i is the mode associated with X_i. X_1 has the lowest mean, and represents the best RMD mode. Without considering dependency between these variables, we can evaluate

this probability as follows:

$$\begin{aligned} Y_i &= X_i - X_1 \\ Y_i &\sim N(\mu_i - \mu_1, \sigma_i^2 + \sigma_1^2) \\ P(X_i < X_1) &= P(Y_i < 0), \end{aligned} \tag{4.10}$$

where $P(Y_i < 0)$ is a familiar normal random variable probability and is computed using *erf* function mentioned in eq. (4.8).

To take into consideration the correlation between the normal random variables and to use the already computed RDO costs, we adopt an iterative approach which combines the two steps of candidates selection and rate-distortion optimization. Based on this approach, we carry out a successive evaluation of the probability that a certain variable is less than the current best RDO cost. Algorithm 4.1 shows the procedure for this. Using this algorithm, we obtain the number of candidate modes, K_L, and the best mode (m_{best}), which is the one with the lowest RDO cost. Based on this algorithm, as soon as one random variable does not satisfy the condition on line 6, we avoid checking this condition for other random variables with larger means because, based on our experiments, computing the extra RDO costs reduces the speedup, and does not improve the quality.

To compute the probability on line 6 of Algorithm 4.1, we show that the normal distributions related to candidate modes are, two by two, bivariate normal and then we use the properties of this joint distribution. To show a bivariate normal distribution for two random variables, many analytical methods are proposed in the literature. However, no best method exists since the results obtained by each of them are different under certain conditions. The most popular methods used to assess multivariate and bivariate normality are Mardia's, Henze-Zirkler's and Royston's, as well as graphical approaches such as Chi-Square Q-Q, perspective and contour plots (Korkmaz *et al.*, 2014). Based on a comprehensive simulation study, presented in (Mecklin and Mundfrom, 2005), on 13 statistical methods for testing the multivariate normal distribution (MVN) with a Monte Carlo study, the authors suggested using Henze-Zirkler's

Algorithm 4.1 RDO best mode selection

Input: $X_i \sim N(\mu_i, \sigma_i^2)$, $\mu_i = f_\mu(C_{RMD_i})$ and
$\sigma_i = f_\sigma(C_{RMD_i})$, $1 \leqslant i \leqslant P_L$, $\mu_i \leqslant \mu_j \; \forall \; i < j$
Output: K_L, m_{best}

```
compute C_RDO1
C_RDObest = C_RDO1
m_best = m_1
K_L = 1
for i = 2 to P_L do
    if P(X_i < C_RDObest) > CL then
        compute C_RDOi
        if C_RDOi < C_RDObest then
            C_RDObest = C_RDOi
            m_best = m_i
        end if
        K_L = i
    else
        break
    end if
end for
return K_L, m_best
```

and Royston's tests because they show better results in terms of error control and power. We therefore use these two tests to show that the RDO costs of the candidate modes, two by two, follow a bivariate normal model. To this end, we apply functions for these tests (Trujillo-Ortiz *et al.*, 2007a,b) to vectors of RDO costs related to blocks with the RMD cost of C_{RMD_i} and C_{RMD_j}; hence, we have two vectors of RDO costs and it is a straightforward procedure to show they are bivariate normal using the referenced functions. Sample results are presented in Table 4.2 for these two tests. The significance level, the *p*-value and the test decision could be interpreted here similarly to what appears in Table 4.1. Similar results are observed for different sequences, with various block sizes and QPs.

Now, we consider the special property of a bivariate normal distribution which states that the observed value of one variable leads to a conditional distribution for the other unobserved one.

Table 4.2 Joint normality test results, *RaceHorses*, 8×8 blocks, QP=32

	Henze-Zirkler	**Royston**
Significance level	0.05	0.05
Test result	MVN verified	MVN verified
p-value	0.051	0.068

The conditional density function for the unobserved variable is:

$$f_{X_j|X_i=x_i}(x_j) = \frac{f_{X_j,X_i}(x_j,x_i)}{f_{X_i}(x_i)}, \tag{4.11}$$

where X_j and X_i are two random variables, among P_L variables, associated with two modes m_j and m_i ($j \neq i$). Substituting the joint normal and normal densities on the right, we have:

$$f_{X_j|X_i=x_i}(x_j) = \frac{1}{\sigma_j\sqrt{1-\rho^2}\sqrt{2\pi}} e^{-\frac{(x_j-(\mu_j+\sigma_j\rho(x_i-\mu_i)/\sigma_i))^2}{2\sigma_j^2(1-\rho^2)}}, \tag{4.12}$$

where ρ is the correlation coefficient, and is obtained from training. Hence, this conditional distribution is normal, with the following mean and variance:

$$\begin{aligned} \mu_{j|X_i=x_i} &= \mu_j + \sigma_j\rho(x_i-\mu_i)/\sigma_i \\ \sigma^2_{j|X_i=x_i} &= \sigma_j^2(1-\rho^2). \end{aligned} \tag{4.13}$$

As can be seen, the conditional mean depends linearly on the observed value, while the conditional variance does not depend on the observation. As a result of this analysis, conditional densities are used for computing the probability on line 6 of Algorithm 4.1, which once again, is a familiar normal random variable probability.

4.1.3 Experimental Results and Discussion

In this section, we present the results for RDO cost modeling to show the contribution of this statistical method in the overall algorithm. To obtain the results, we use the HEVC test model HM 15.0 and a PC equipped with an Intel® Core™ i7-4790 CPU @ 3.60 GHz

and 32 GB of RAM. The configuration and profile are set to *all-intra* and *Main profile*, respectively. We report the results, using the common test conditions recommended in (Bossen, 2013), compared to HM based on time reduction (TR), Bjøntegaard delta rate (Bjøntegaard, 2001) (BD-Rate) and Bjøntegaard delta peak signal-to-noise ratio (Bjøntegaard, 2001) (BD-PSNR). These conditions include video sequences in different classes (A (2560×1600), B (1920×1080), C (832×480), D (416×240) and E (1280×720)) to cover various applications and resolutions. The results are averaged over four QPs: 22, 27, 32 and 37; they therefore cover low and high bitrate scenarios. Time reduction is defined as:

$$TR(\%) = \frac{T_{Proposed} - T_{HM}}{T_{HM}} \times 100\%, \tag{4.14}$$

where $T_{Proposed}$ and T_{HM} are the total encoding time of the proposed encoder and HM encoder, respectively. To calculate BD-Rate and BD-PSNR, we apply Bjøntegaard algorithm by using the following combined PSNR (Sze *et al.*, 2014):

$$\text{PSNR} = \frac{6 \cdot \text{PSNR}_\text{Y} + \text{PSNR}_\text{U} + \text{PSNR}_\text{V}}{8}. \tag{4.15}$$

This assures us that we measure the impact of the proposed algorithms on the total quality. To achieve preliminary results in this section and sections 4.2.2, 4.3.2, 4.4.3, 5.1.3 and 5.2.2, we use the first hundred frames of the recommended sequences while for the final results in sections 4.5 and 5.3.4, all frames of the sequences are used.

Table 4.3 shows the results while only RDO cost modeling and candidates selection are implemented (see fig. 4.1). We observe that RDO cost modeling, one of our main contributions, leads to a time reduction of nearly 30% with a BD-Rate increase of 0.8%; a trade-off which is quite appealing. To derive the models (normal distributions), the *RaceHorses* sequence is used as the training sequence, and hence, the average results are presented with and without considering this sequence. This approach of presenting the results when a training sequence is selected among the common test sequences was adopted in (Zhang *et al.*, 2017a). Based on this training, the confidence level (CL) on line 6 of the Algorithm 4.1 is set to 0.2, which represents

a very good trade-off between complexity reduction and coding efficiency. Depending on the application, CL can change to have a faster or higher quality encoder. Also, the correlation coefficient ρ is computed based on the training data and is averagely 0.48. To obtain ρ, for blocks of a specific RMD cost (e.g. C_{RMD_i}), we associate a vector of RDO costs with distributions X_i and similarly we have another vector of RDO costs with distributions X_j associated to blocks with RMD cost of C_{RMD_j}. ρ_{ij} of these two distributions X_i and X_j is obtained by computing the Pearson correlation coefficient between these two vectors.

Table 4.3 Experimental results while implementing RDO cost modeling compared to HM, first hundred frames

Class	Video sequence	TR (%)	BD-Rate (%)	BD-PSNR (dB)
A	Traffic	-29.2	1.16	-0.055
	PeopleOnStreet	-32.4	1.23	-0.060
	Nebuta	-31.7	0.37	-0.023
	SteamLocomotive	-30.9	0.39	-0.015
B	Cactus	-29.6	0.74	-0.024
	Kimono	-30.2	1.49	-0.049
	ParkScene	-29.1	0.81	-0.031
	BasketballDrive	-31.1	0.92	-0.024
	BQTerrace	-30.0	0.45	-0.023
C	BQMall	-30.1	0.72	-0.038
	PartyScene	-24.7	0.48	-0.032
	RaceHorsesC	-26.8	0.48	-0.027
	BasketballDrill	-30.9	0.53	-0.024
D	RaceHorses	-28.4	0.77	-0.044
	BasketballPass	-29.9	0.77	-0.041
	BlowingBubbles	-27.0	0.57	-0.030
	BQSquare	-24.1	0.54	-0.039
E	FourPeople	-31.6	1.19	-0.061
	Johnny	-31.3	1.30	-0.049
	KristenAndSara	-30.0	1.28	-0.060
Average (with training sequence)		**-29.4**	**0.80**	**-0.038**
Average (without training sequence)		**-29.5**	**0.81**	**-0.037**

4.2 Fast Chroma Mode Decision

4.2.1 Proposed Method

For chroma mode decision, we propose a mode classification which categorizes five chroma modes into two groups of promising and non-promising candidates, and excludes non-promising ones from the RDO. However, we always add the best luma mode to the group of promising candidates if it is not already included in the list. Figure 4.5 shows the chroma mode decision of the proposed method. The following presents the proposed method in detail.

HEVC follows a 4:2:0 chroma sampling format, which means each CTU of size $N \times N$ includes one luma CTB of size $N \times N$ and two chroma CTBs of size $N/2 \times N/2$. Similarly, CUs, PUs and TUs include one luma block (CB, PB, TB) of size $N \times N$ and two chroma blocks of size $N/2 \times N/2$. As we mentioned in section 2.4, for both chroma components, HEVC presents five prediction modes, including the corresponding luma mode. Table 4.4 shows these five modes. In this table, X indicates luma modes other than planar (0), dc (1), vertical (26) or horizontal (10). Also, 34 indicates a diagonal mode.

Table 4.4 Chroma intra modes

	Luma intra mode				
	0	26	10	1	X
Chroma mode (0)	34	0	0	0	0
Chroma mode (1)	26	34	26	26	26
Chroma mode (2)	10	10	34	10	10
Chroma mode (3)	1	1	1	34	1
Chroma mode (4)	0	26	10	1	X

To reduce chroma mode decision complexity, we propose a fast mode selection. Five different selection methods are examined, which reduce the number of modes going through the RDO process, and are compared to one another. These methods are as follows:

A) Using only the best luma mode

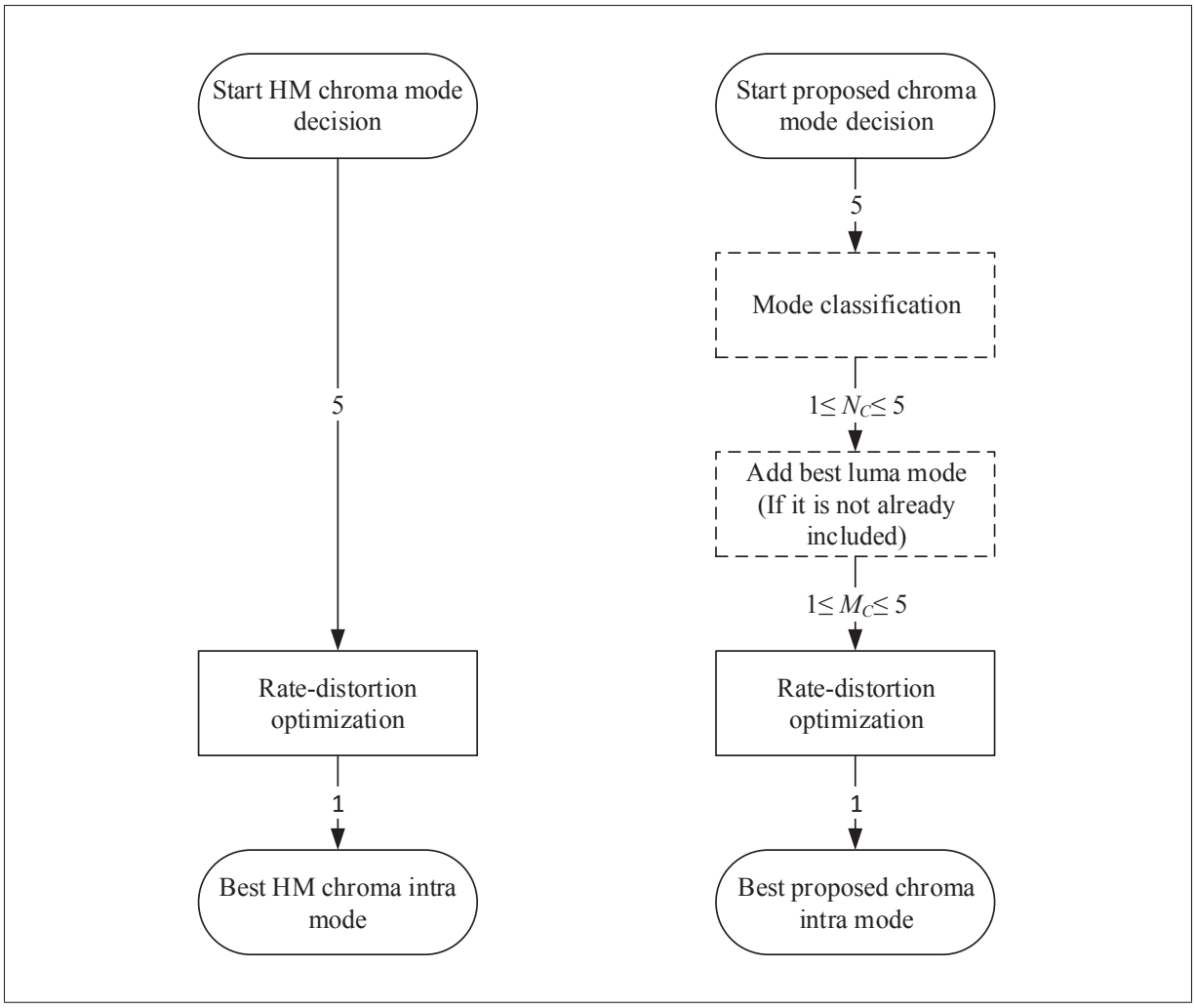

Figure 4.5 Block diagram of chroma mode decision at PU level for HM (left) and proposed method (right). Arrows show the number of modes in each step.

B) Using only the mode with the lowest SATD cost (SATD mode)

C) Using the mode with the lowest SATD cost and adding the best luma mode (SATD mode + luma mode)

D) Selecting a short list of candidate modes if their SATD costs are clearly separated by a gap from other candidates' costs (mode classification)

E) Selecting a short list of candidate modes by finding a gap and adding the best luma mode if it is not already included (mode classification + luma mode)

Figure 4.6 shows how these five methods select the chroma mode (CM), among five candidates, which is the mode the encoder uses to obtain the residual for the current chroma PB. Method A simply chooses the same mode as the selected luma mode (LM). For other methods, first SATD costs of the five candidates are computed to form the set $\Psi_1 = \{\mathrm{CM}_0, \mathrm{CM}_1, \mathrm{CM}_2, \mathrm{CM}_3, \mathrm{CM}_4\}$ which includes the candidates in increasing order of their SATD cost. SATD cost is obtained based on the transformed distortion between the current PB and its predicted PB by applying the associated candidate mode. By using method B, CM_0 which is the candidate with the least SATD cost is selected as the final chroma mode. In method C, the encoder selects the mode with the least RDO cost between CM_0 and LM. For methods D and E, a mode classification step is proposed to select a subset of candidates for RDO process. Based on experimental results, this idea proved to be a very good strategy for chroma components. In this approach, we look for a manifest distance between SATD costs of two consecutive candidates from set Ψ_1. Then, based on this distance the candidates are categorized into two groups of promising and non-promising and we exclude the non-promising ones from further processing. The distance is defined as:

$$d = \alpha \times (Cost_{max} - Cost_{min}), \tag{4.16}$$

where α is an adjustable parameter, trading off computational complexity and visual quality and, for N candidates $(N > 2)$, it is constrained to the following condition:

$$\frac{1}{N-1} < \alpha < 1. \tag{4.17}$$

Algorithm 4.2 shows the procedure for mode classification which forms set Ψ_2 from set Ψ_1. Line 5 of the algorithm removes candidate CM_j from Ψ_2. In addition, Table 4.5 shows the number of chroma modes, M_C, which are used in the RDO process to find the best mode. M_C is adaptive for the last two methods since when we are looking for a distance between

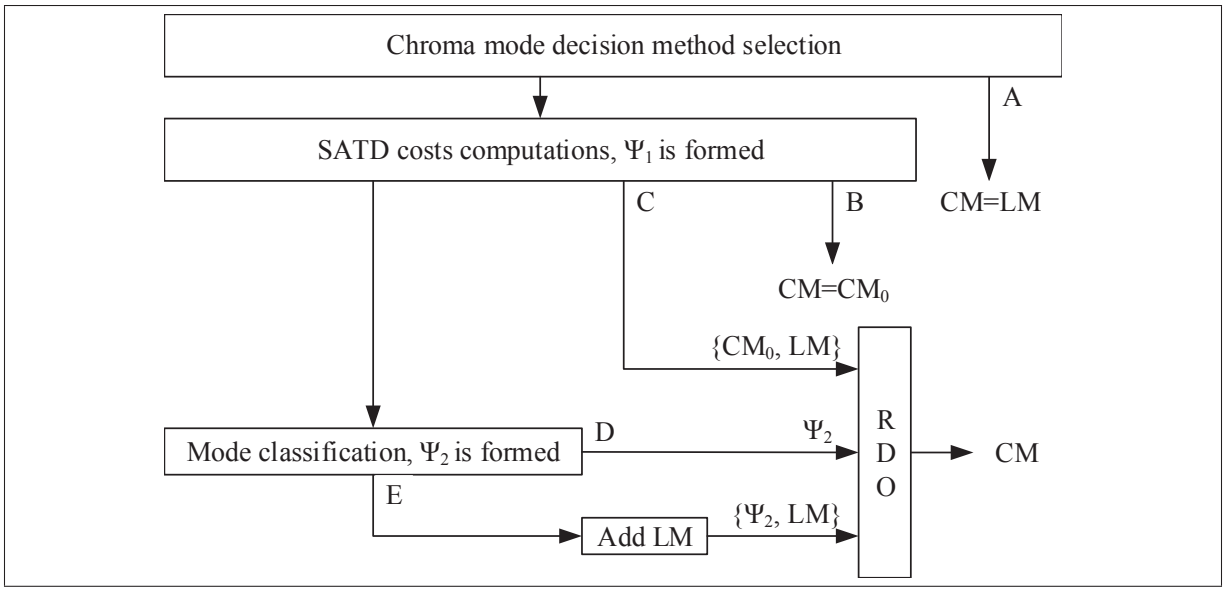

Figure 4.6 Chroma mode decision methods.

Algorithm 4.2 Chroma mode classification

Input: $\Psi_1 = \{\mathrm{CM}_0, \mathrm{CM}_1, \mathrm{CM}_2, \mathrm{CM}_3, \mathrm{CM}_4\}$
Output: Ψ_2

```
Ψ2 = Ψ1
for i = 0 to 3 do
    if (Cost(CM_{i+1}) − Cost(CM_i)) > d then
        for j = i + 1 to 4 do
            Ψ2 = Ψ2 \ {CM_j}
        end for
        break
    end if
end for
return Ψ2
```

SATD costs, the number of modes selected for the RDO process depends on the block context. Experimental results for these five different methods are provided in section 4.2.2.

Table 4.5 Number of selected chroma modes

Chroma method	M_C
A	1
B	1
C	2
D	$1 \leqslant M_C \leqslant 5$
E	$1 \leqslant M_C \leqslant 5$

4.2.2 Experimental Results and Discussion

In this section, we compare the five chroma mode decision approaches, presented in the last section, based on quality and complexity reduction. The implementation conditions and configuration are as it is mentioned in section 4.1.3. Table 4.6 shows the results in terms of time reduction, BD-Rate and BD-PSNR for the entire encoder compared to the HM, while implementing the proposed chroma methods instead of HM standard chroma implementation. Based on these experiments, and considering the trade-off between complexity reduction and coding efficiency, we believe the best results are achieved by using a combination of the mode classification and the best mode of the luma component (method E), since this results in significant speedup without noticeably affecting the rate-distortion performance. The results show that there is almost no quality loss using this proposed chroma method. However, a different method may be selected to achieve faster processing, with a penalty in terms of the quality. To show the contribution of chroma mode decision in the overall mode decision, Table 4.7 presents the results for all sequences using method E. The experiments also show that chroma intra prediction is a complex component of the entire encoder, and that more research is required in the area of intra chroma complexity reduction.

Table 4.6 Experimental results for the proposed chroma mode decision methods, modified HM (chroma methods implemented) compared to standard HM

Video sequence	**Chroma method**	**TR (%)**	**BD-Rate** (%)	**BD-PSNR (dB)**
RaceHorses	A	-12.6	1.75	-0.101
	B	-10.0	4.24	-0.246
	C	-7.6	0.21	-0.013
	D ($\alpha = 1/2$)	-4.5	0.22	-0.013
	E ($\alpha = 5/16$)	-6.9	0.11	-0.006
BasketballPass	A	-12.0	2.15	-0.112
	B	-10.0	5.63	-0.290
	C	-10.0	0.19	-0.010
	D ($\alpha = 1/2$)	-4.7	0.12	-0.006
	E ($\alpha = 5/16$)	-6.2	0.03	-0.002
BlowingBubbles	A	-9.9	1.36	-0.070
	B	-10.1	3.18	-0.165
	C	-8.0	0.21	-0.010
	D ($\alpha = 1/2$)	-3.9	0.18	-0.009
	E ($\alpha = 5/16$)	-6.2	0.05	-0.002

Table 4.7 Experimental results while implementing fast chroma mode decision (method E) compared to HM, first hundred frames

Class	**Video sequence**	**TR (%)**	**BD-Rate** (%)	**BD-PSNR (dB)**
A	Traffic	-7.1	0.04	-0.002
	PeopleOnStreet	-6.6	0.04	-0.002
	Nebuta	-7.1	0.01	-0.001
	SteamLocomotive	-5.2	0.02	-0.001
B	Cactus	-5.9	0.07	-0.002
	Kimono	-5.9	0.06	-0.002
	ParkScene	-5.5	0.06	-0.002
	BasketballDrive	-7.6	0.11	-0.003
	BQTerrace	-5.2	0.05	-0.003
C	BQMall	-5.8	0.07	-0.004
	PartyScene	-5.8	0.09	-0.006
	RaceHorsesC	-5.7	0.05	-0.003
	BasketballDrill	-6.8	0.11	-0.005
D	RaceHorses	-6.9	0.11	-0.006
	BasketballPass	-6.2	0.03	-0.002
	BlowingBubbles	-6.2	0.05	-0.002
	BQSquare	-5.8	0.08	-0.005
E	FourPeople	-5.3	0.06	-0.003
	Johnny	-4.5	0.09	-0.004
	KristenAndSara	-5.3	0.12	-0.005
Average		**-6.0**	**0.07**	**-0.003**

4.3 Low Complexity Edge Detection for Mode Decision

4.3.1 Proposed Method

Although RMD is less complex than RDO, it still consumes a considerable amount of time, and needs to be replaced by a less complex process. To this end, we apply a gradient-based edge detection approach to determine the main directions in a block and exclude irrelevant angular modes from further processing. A gradient-based approach is useful since intra coding uses objects' edges to find the best mode (direction) for forming the prediction block. However, there are two problems with edge detection algorithms in intra mode decision. First, as the objects' edges are not exactly aligned with HEVC intra modes' directions, an edge detection algorithm with unique output cannot predict the intra mode ideally. Therefore, to use gradient-based algorithms efficiently, we propose an enhanced gradient-based mode detector. For any detected edge, we assign weights to a few adjacent modes to obtain an improved accuracy for the method. As the second shortcoming of edge detection algorithms, they fail to predict the final best intra mode for a given block. Nonetheless, they are very good for excluding irrelevant directional modes. Considering this, and the fact that they are less complex than other methods such as RMD, we consider edge detection as a mode excluding step of the luma mode decision method.

In the standard implementation of HEVC, the coding process for PUs starts at the CTU level. From there, the process tests all combinations of PUs and all modes for each of them at different depths. Finally, the best depth, the best PU sizes and the best modes for each PU are selected for the CTU. With this in mind, we determine gradients for each pixel at the CTU level. This allows us to use the edge information for each pixel at any depth and prevent a repetition of the calculations for each PU. Using the gradient, we are able to determine the directions with maximum variation of pixel values. The picture edges are perpendicular to these directions, and they show the dominant angular modes for intra prediction.

To compute the gradient, we investigate Sobel, Scharr, Prewitt and Roberts cross as computationally light and popular operators used to compute the approximation of the gradient. Sobel is an isotropic 3×3 image gradient operator which computes an approximation of the gradient of the intensity function at each pixel of the image. Since the operation works by convolving an integer-valued and small filter with the image, the computations are relatively inexpensive. Horizontal and vertical kernels, which are used to calculate approximation of the directional derivatives, can be expressed as follows:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}. \tag{4.18}$$

Another operator proposed in the literature for gradient computation is Scharr, which applies different multipliers from Sobel, and its kernels are expressed as follows:

$$G_x = \begin{bmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{bmatrix} \qquad G_y = \begin{bmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ 3 & 10 & 3 \end{bmatrix}. \tag{4.19}$$

Prewitt is yet another discrete differentiation operator used to approximately compute the gradient of the intensity function of an image, which uses an integer-valued and small operator to convolve with the image. As a result, this operator, like Sobel, is an inexpensive one in terms of computations. The Prewitt operator uses the following 3×3 kernels:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}. \tag{4.20}$$

In contrast to the above operators, Roberts cross approximates the gradient of an image by using the following 2×2 kernels:

$$G_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad G_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}. \tag{4.21}$$

Since the kernels are diagonal, variations are computed in a diagonal direction by this operator, which is the most useful in terms of detecting diagonal edges. The most interesting aspect of this operator is its simplicity, as its kernels include only two pixels for gradient computation, while Sobel, Scharr and Prewitt engage six pixels.

To choose the most suitable operator, for our problem, we look for the related features such as low complexity and accuracy. It needs to be faster, as compared to the RMD, and it needs to exclude only those directional modes which are unlikely to be the best mode. To this end, in section 4.3.2, we compare these operators based on their accuracy in order to include the best mode in the candidates list; we also carry out a comparison based on the time reduction the operators provide.

For all operators, and at each pixel, the gradient magnitude is computed using:

$$G = \sqrt{G_x^2 + G_y^2}. \tag{4.22}$$

In addition, the gradient components are used to calculate the gradient direction as follows:

$$\Theta = \text{atan}\left(\frac{G_y}{G_x}\right). \tag{4.23}$$

To avoid the computationally expensive and inefficient square-root operation, it is advisable to estimate the gradient magnitude by $|G_x| + |G_y|$ or $\max\{|G_x|, |G_y|\}$ (Ziou *et al.*, 1998). We use the former in this thesis. Moreover, we use G_y/G_x instead of $\text{atan}(G_y/G_x)$ to avoid the

resource-intensive Arctan operation. Using these estimations results in less time consumption, while the quality is affected insignificantly.

As it is mentioned, the gradient magnitude and direction are computed for each pixel at the CTU level, and this data is passed to all PUs inside the CTU. At the PU level, depending on the gradient direction, a main mode is associated with each pixel. The main mode is defined as the closest mode corresponding to the edge. This mode is achieved by comparing G_y/G_x to predefined limits that are pre-calculated based on the specified angles for intra modes in the HEVC standard. The high and low limits for each mode are shown in Table 4.8. To be more accurate in finding modes that provide the best prediction, two adjacent modes are considered, in addition to the main mode. This avoids considering only the main mode, as the edge is rarely perfectly aligned with it. Therefore, we also give weights to modes adjacent to the main mode based on the direction of the detected edge, as it is illustrated in fig. 4.7. The weights that are given to each of these modes are calculated as follows:

$$W_{main} = |G_x| + |G_y|, \tag{4.24}$$

$$WF_P = \left(\text{atan}(L_H) - \text{atan}\left(\frac{G_y}{G_x}\right)\right) / (\text{atan}(L_H) - \text{atan}(L_L)), \tag{4.25}$$

$$WF = \left(L_H - \frac{G_y}{G_x}\right) / (L_H - L_L), \tag{4.26}$$

$$W_{adjacent1} = (1 - WF) \times (|G_x| + |G_y|), \tag{4.27}$$

$$W_{adjacent2} = WF \times (|G_x| + |G_y|), \tag{4.28}$$

where W_{main}, $W_{adjacent1}$ and $W_{adjacent2}$ indicate main mode and adjacent modes weights. L_H and L_L are the high and low limits, respectively and WF_P is the perfect form of the mode weight factor which gives weights of 0.5 for two adjacent modes if the edge sits exactly on a mode. However, as previously mentioned, computing the *atan* function is avoided, and an approximation is used (WF). It should be noted that the horizontal mode (mode 10) is considered as a special case, and the weight factor for this mode is defined as follows:

$$WF_{10} = 0.5\left(1 + \frac{40.73548}{|G_y/G_x|}\right) \tag{4.29}$$

This is because for this mode, there is infinity for the low limit or high limit and the normal formula cannot be applied in this situation. Following these computations, we accumulate the weights of the modes by traversing all pixels of the PU, while each pixel increases the weight of three modes. Finally, we obtain a histogram that indicates the accumulated weights of directional modes for the whole PU, where we consider N_L most powerful modes (see fig. 4.1) as the best candidates selected by edge detection and exclude the remaining candidates from further processing.

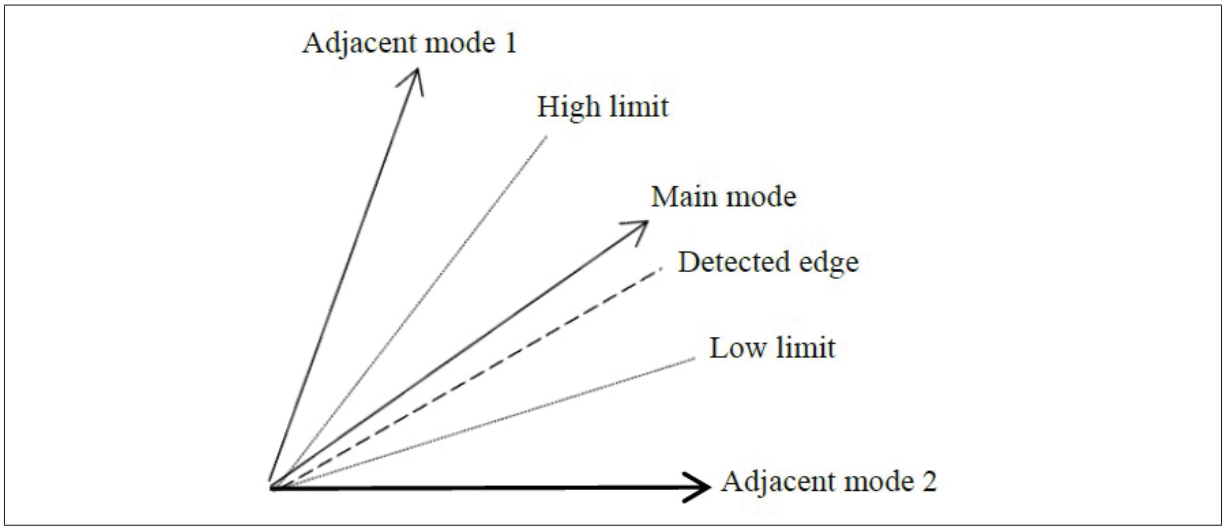

Figure 4.7 Detected edge and three related modes.

Table 4.8 High and low limits of G_y/G_x for angular modes

Mode	lowLimit	highLimit	Mode	lowLimit	highLimit
2	-1.15928	-1	18	0.86261	1.15928
3	-1.53711	-1.15928	19	0.65057	0.86261
4	-1.98666	-1.53711	20	0.50336	0.65057
5	-2.59240	-1.98666	21	0.38574	0.50336
6	-3.61354	-2.59240	22	0.27674	0.38574
7	-5.76314	-3.61354	23	0.17352	0.27674
8	-11.61240	-5.76314	24	0.08611	0.17352
9	-40.73548	-11.61240	25	0.02455	0.08611
10	$-\infty$	-40.73548	26	-0.02455	0.02455
10	40.73548	∞	27	-0.08611	-0.02455
11	11.61240	40.73548	28	-0.17352	-0.08611
12	5.76314	11.61240	29	-0.27674	-0.17352
13	3.61354	5.76314	30	-0.38574	-0.27674
14	2.59240	3.61354	31	-0.50336	-0.38574
15	1.98666	2.59240	32	-0.65057	-0.50336
16	1.53711	1.98666	33	-0.86261	-0.65057
17	1.15928	1.53711	34	-1	-0.86261

4.3.2 Experimental Results and Discussion

In this section, we present the results achieved for fast mode decision based on edge detection and gradient analysis. The implementation conditions and configuration are as mentioned in section 4.1.3. First, we compare the four gradient operators, mentioned in the last section, based on accuracy and complexity to select the best one for our method and then we provide time reduction achieved by applying a gradient-based algorithm to the encoder. To this end, Table 4.9 shows the error rate of each operator (i.e., the rate at which the best mode is excluded from the candidates list) for the following two configurations:

A) Test model where RMD is replaced by a gradient operator and MPMs are not added;

B) Test model where RMD is replaced by a gradient operator (MPMs are added).

In addition, Table 4.10 shows the results of the entire algorithm, including the RDO cost prediction and chroma mode decision approaches (discussed in sections 4.1 and 4.2) while

Table 4.9 Error rate of gradient operators, Config. A and Config. B

		Error (%)	
Video sequence	**Kernel**	**Config A**	**Config B**
RaceHorses	Sobel	24.7	11.9
	Prewitt	24.9	11.8
	Scharr	24.8	12.1
	Roberts cross	27.1	13.1
BasketballPass	Sobel	26.6	7.9
	Prewitt	27.3	8.2
	Scharr	26.9	8.1
	Roberts cross	29.3	8.8
BlowingBubbles	Sobel	23.7	11.5
	Prewitt	23.9	11.7
	Scharr	24.2	11.8
	Roberts cross	27.5	13.6

different gradient kernels are implemented for three sequences with different video textures. In the same way, Table 4.11 presents the results averaged over all sequences. Based on these results, Prewitt provides the best trade-off between time reduction and bit-rate increment, and is selected as our operator for the gradient analysis section. However, considering the time reduction table and error table, the difference between theses operators is not very significant, mostly because we apply the gradient analysis at the CTU level, and not at the PU level. This allows us to compute the gradient just once for each pixel, and to use this information for each PU which contains that pixel. To demonstrate the contribution of the gradient analysis in the overall algorithm, results for all test sequences while only gradient analysis is implemented are shown in Table 4.12. We observe that gradient analysis provides a time reduction of 11.4% with a BD-Rate increase of 0.62%.

Table 4.10 Experimental results of the proposed method, implementing different gradient operators, compared to HM

Video sequence	Kernel	TR (%)	BD-Rate (%)
RaceHorses	Sobel	-44.3	1.25
	Prewitt	-46.0	1.20
	Scharr	-42.2	1.21
	Roberts cross	-46.1	1.29
BasketballPass	Sobel	-47.1	1.70
	Prewitt	**-47.4**	**1.66**
	Scharr	-47.4	1.78
	Roberts cross	-46.2	1.71
BlowingBubbles	Sobel	-43.1	1.06
	Prewitt	**-43.7**	**1.05**
	Scharr	-43.2	1.05
	Roberts cross	-43.2	1.15

Table 4.11 Average results over all recommended sequences (Bossen, 2013) of the proposed method compared to HM using different gradient operators

Kernel	TR (%)	BD-Rate (%)	BD-PSNR (dB)
Prewitt	-47.2	1.36	-0.062
Sobel	-47.1	1.38	-0.063
Scharr	-47.0	1.40	0.064
Roberts cross	-47.4	1.60	-0.073

Table 4.12 Experimental results while implementing gradient analysis (Prewitt) compared to HM, first hundred frames

Class	Video Sequences	TR (%)	BD-Rate (%)	BD-PSNR (dB)
A	Traffic	-12.0	0.46	-0.021
	PeopleOnStreet	-11.5	0.64	-0.031
	Nebuta	-10.9	0.36	-0.022
	SteamLocomotive	-13.9	0.28	-0.011
B	Cactus	-11.9	0.78	-0.026
	Kimono	-14.1	0.45	-0.015
	ParkScene	-11.4	0.40	-0.016
	BasketballDrive	-11.9	1.47	-0.037
	BQTerrace	-10.3	0.44	-0.021
C	BQMall	-8.8	0.56	-0.029
	PartyScene	-9.7	0.51	-0.034
	RaceHorsesC	-11.1	0.35	-0.019
	BasketballDrill	-11.9	0.36	-0.016
D	RaceHorses	-10.4	0.49	-0.028
	BasketballPass	-10.8	0.94	-0.051
	BlowingBubbles	-10.0	0.43	-0.023
	BQSquare	-9.1	0.77	-0.056
E	FourPeople	-11.9	0.70	-0.036
	Johnny	-12.6	0.93	-0.036
	KristenAndSara	-12.9	1.06	-0.049
Average		**-11.4**	**0.62**	**-0.029**

4.4 Fast Mode Decision Based on SATD Cost Classification

In this section, we introduce a fast mode decision approach which is presented in (Jamali *et al.*, 2015). In addition, a system is designed based on this approach which is the subject of a patent application (Jamali *et al.*, 2014). This approach is based on an improved edge detection, consideration of most relevant modes from neighboring blocks, and a classification of SATD costs permitting the elimination of several candidate modes prior to rate distortion optimization. At the first step, the edge detection is applied as described in section 4.3. Then, we add DC and planar modes and also the most relevant modes to employ neighboring blocks for mode decision. Next, we select *N* best modes and carry out a binary classification, based on SATD costs, and eliminate less promising candidates. This step significantly reduces the number of candidates to be processed by RDO, providing considerable time reduction. Finally, in RDO dodging, to achieve further time savings, we bypass the RDO process completely when the mode with the lowest SATD cost meets certain specific conditions. We integrate all these steps in one general method which is illustrated as a block diagram in fig. 4.8. Following, we describe each block in details.

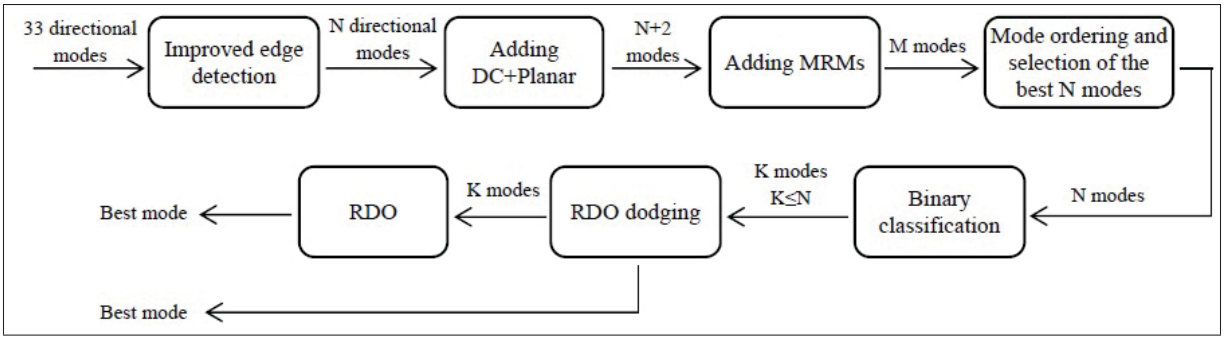

Figure 4.8 Block diagram of the proposed SATD cost classification algorithm.

4.4.1 Most Relevant Modes of the Neighboring Blocks

In this section, we use the spatial correlation across a frame to reduce the number of likely modes to be the best mode of the current block. To use the spatial correlation, the context

in the current frame around the current block is exploited. This is in contrast to temporal correlation where the context comes from other frames either before or after the current frame. Spatial correlation is used in mode decision by considering the modes of the neighboring blocks as potential candidates to be the best mode of the current block. We use this concept in our method after edge detection to include the neighboring blocks' modes to the list of candidates. Combining edge detection by approaches introduced in this section makes sure that really potential candidates are selected for further processing.

There are two improvement proposed here to enhance using neighboring blocks for intra mode decision. Firstly, while previous works just consider two top and left blocks for this purpose, in this work we consider five top, left, top-left, top-right and down-left blocks to increase the accuracy. Secondly, in contrary to other works which consider the neighboring modes in any case, we add the modes of the neighboring blocks to the set of potential modes just if they are relevant. By relevancy, we mean if they satisfy some condition; i.e., if based on their direction, they are likely to be the best mode of the current block. For example, if the mode of the top block is vertical or near-vertical, then it is added to the list; otherwise, if it is horizontal or near-horizontal, it is less likely to be the best mode of the current block and is not considered. We do the same for the left block if its mode is horizontal or near-horizontal. Similarly, the top-right, top-left and down-left blocks are processed in the same manner. For each of these neighboring blocks, $2n+1$ modes are considered, which we call most relevant modes (MRMs), and if a neighboring block's mode corresponds to one of these modes, then it is considered for the further processing.

Figure 4.9 shows the concept of the most relevant modes of the neighboring blocks. For example, for the top-left block with $n = 2$ the MRMs are modes 16, 17, 18, 19 and 20, and if that block selects one of these modes, then the mode is considered as a candidate for the current block. In MRM step, a minimum of zero and a maximum of five modes are added to the set of potential candidates. In addition to the neighboring modes, DC and planar modes are added after edge detection, since the edge detection algorithms do not consider the non-directional modes. If the number of selected candidates after edge detection is N, then after adding the

MRMs, DC and planar the set of potential candidates has M members (see fig. 4.8), where:

$$N+2 \leq M \leq N+7 \tag{4.30}$$

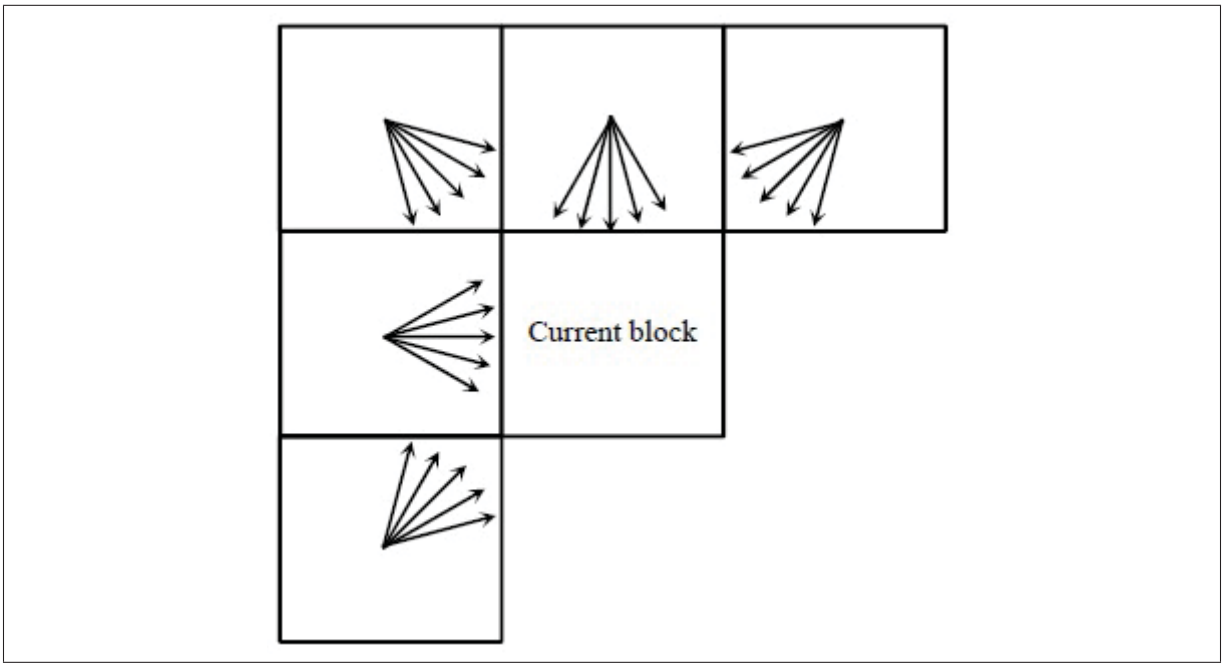

Figure 4.9 Most relevant modes of the neighboring blocks.

4.4.2 Mode Ordering, Binary Classification and RDO Dodging

In this section, we propose a mode classification method for luma mode decision similar to the method presented in section 4.2. Based on this method, we are looking for a dominant gap among the SATD costs. If such a gap exists, the modes with SATD cost lower than the gap are selected as the promising candidates and the other modes are excluded from further processing. Based on our experiments with different video sequences, this approach leads to a high time reduction while the quality loss is negligible. Thus after adding MRMs, we order the selected M modes from the previous step based on the SATD costs, from the lowest cost to the highest cost, and select the first N modes (see fig. 4.8). The modes are then classified into two classes:

powerful contenders with lower costs and weak ones with higher costs. We keep the powerful modes and pass them on to the next step and exclude the weak modes from further processing. This classification could be based on different criteria, but we have found that if a dominant distance exists among the SATD costs, we can efficiently remove the modes with higher costs from the very time-consuming RDO process without affecting the rate-distortion performance. In other words, it is wasteful to test candidates whose costs are much higher than others'. For example, if two candidates have very low costs while the others have significantly higher costs, it is often a waste of time to evaluate all candidates by the computationally expensive RDO, as only the two low-cost candidates usually win. Figure 4.10 shows a detected gap among SATD costs where C_{min} is the lowest cost and C_{max} is the highest cost. Finding such a gap is very helpful, and significantly decreases the time consumption for mode decision in intra coding. It reduces the number of modes to K, where $K \leq N$ (see fig. 4.8). $K = N$ occurs when no gap is found. This gap could be a fixed one or could be adaptive based on the block size, quantization parameter, or minimum and maximum cost values or any other criterion. For obtaining results in this section, we consider adaptivity based on block size, and the difference between the maximum and minimum cost values. The gap is defined as follows:

$$g = \alpha \times (C_{max} - C_{min}), \tag{4.31}$$

where $\alpha \leq 1$ is an empirical parameter, and is adjusted based on the block size (see section 4.4.3).

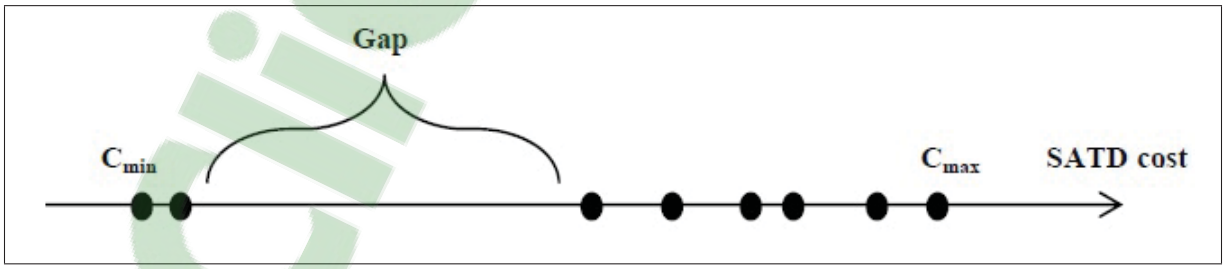

Figure 4.10 A gap among SATD costs.

Algorithm 4.3 illustrates how this method works. In this algorithm, Ψ is the set of selected candidates before applying mode classification which is obtained using the following set:
$S = \{dc + planar + MRMs + selected\ modes\ based\ on\ edge\ detection\} = \{m_1, m_2, \ldots m_M\}$.

Then, SATD costs are computed for these M modes and Ψ is generated based on these costs:
$\Psi = \{N\ modes\ with\ lowest\ costs\ in\ S\} = \{m_1, m_2, \ldots m_N\}$.

At the end, promising modes form a set called P as follows:
$P = \{promising\ modes\ based\ on\ gap\ analysis\} = \{m_1, \ldots m_K\}$.
While
$P \subseteq \Psi \quad and \quad K \leq N$

P is the set of modes that are selected to be investigated by the RDO process.

Algorithm 4.3 Selecting the promising modes based on SATD cost classification

Input: Ψ *(set of N best modes selected by edge detection and MRMs)*
Output: P *(set of K promising modes to be processed by RDO)*
1 *Sort the modes of* Ψ *from the lowest to the highest SATD cost and obtain* C_{min} *and* C_{max}
2 $P = \Psi$
3 $g = \alpha \times (C_{max} - C_{min})$
4 **for** $i = 1$ to $N - 1$ **do**
5 **if** $(SATD(m_{i+1}) - SATD(m_i)) > g$ **then**
6 **for** $j = i + 1$ to N **do**
7 $P = P \backslash m_j$
8 **end for**
9 **break**
10 **end if**
11 **end for**
12 **return** P

Before RDO process, another step is applied called RDO dodging, which again exploits the modes from five neighboring blocks to bypass the RDO process under certain conditions. In this approach, if the mode with the lowest SATD cost is one of the most relevant modes, then it is highly likely to be the best intra mode for the current block. For instance, if the lowest cost candidate for the current block is the vertical mode and the best mode of the above block

is also vertical, we choose this mode as the final decision, and RDO is omitted. To create the set of most relevant modes at this step, we consider $2m+1$ modes from neighboring blocks similar to section 4.4.1. If the candidate with the lowest cost after ordering and classification is one of these relevant modes, it is selected as the best mode; otherwise, all K candidates are investigated by RDO and the candidate with lowest RDO cost is selected as the final best mode.

4.4.3 Experimental Results and Discussion

The proposed SATD cost classification and other modules shown in fig. 4.8 are implemented in the HM 15.0. The implementation platform is an Intel® i7-3770 CPU-3.40, 12 GB of RAM, running Windows 7. We used the recommended sequences in (Bossen, 2013) to implement our proposed algorithm. The HM is configured in *All-Intra* mode, and is executed for quantization parameters of 22, 27, 32 and 37. The parameters of the algorithm are: $N = 8$ and $\alpha = 1/4$ for block sizes of 4×4 and 8×8 and $N = 3$ and $\alpha = 2/3$ for block sizes of 16×16, 32×32 and 64×64. Also, n (section 4.4.1) and m (section 4.4.2) are set to 3 and 1 for selecting the most relevant modes. Table 4.13 shows the results of experiments in terms of BD-Rate, BD-PSNR_Y and time reduction (TR), in comparison to HM. Using Bjontegaard metrics, the table shows the average differences in rate-distortion performance. The time reduction is calculated for each QP, and the average over all QPs is presented in the table. According to the experimental results, we achieve an average of 35.6% time reduction over all video sequences, and up to 39.2% in comparison with the anchor implementation. The main contributors to this time reduction are a decrease in the number of modes for SATD calculations in the RMD step and a decrease in the number of modes in the RDO process. The quality loss is, on average, a 1.07% increase in the BD-Rate, which only slightly affects the rate-distortion performance. To justify this, fig. 4.11 shows the RD curves of the proposed algorithm versus HM for the *RaceHorses* sequence. From this figure, we can see that both implementations have almost the same rate-distortion performance.

Table 4.13 Experimental results of the proposed method compared to HM, first hundred frames

Class	Video sequence	TR (%)	BD-Rate (%)	BD-PSNR$_Y$ (dB)
A	Traffic	-35.4	0.95	-0.051
	PeopleOnStreet	-34.1	1	-0.057
	Nebuta	-34.1	0.53	-0.039
	SteamLocomotive	-37.8	0.48	-0.025
B	Cactus	-36.1	1.34	-0.05
	Kimono	-39.2	0.79	-0.028
	ParkScene	-37.5	0.87	-0.039
	BasketballDrive	-38.4	2.17	-0.059
	BQTerrace	-35.2	0.79	-0.048
C	BQMall	-34.3	1.15	-0.068
	PartyScene	-32.8	1.18	-0.092
	RaceHorsesC	-34.7	0.72	-0.047
	BasketballDrill	-33.1	0.8	-0.039
D	RaceHorses	-34.1	0.99	-0.065
	BasketballPass	-36	1.45	-0.085
	BlowingBubbles	-33.7	1.01	-0.06
	BQSquare	-32.7	1.38	-0.123
E	Vidyo1	-36.8	1.31	-0.066
	Vidyo3	-37.4	1.23	-0.069
	Vidyo4	-37.7	1.33	-0.061
Average		**-35.6**	**1.07**	**0.059**

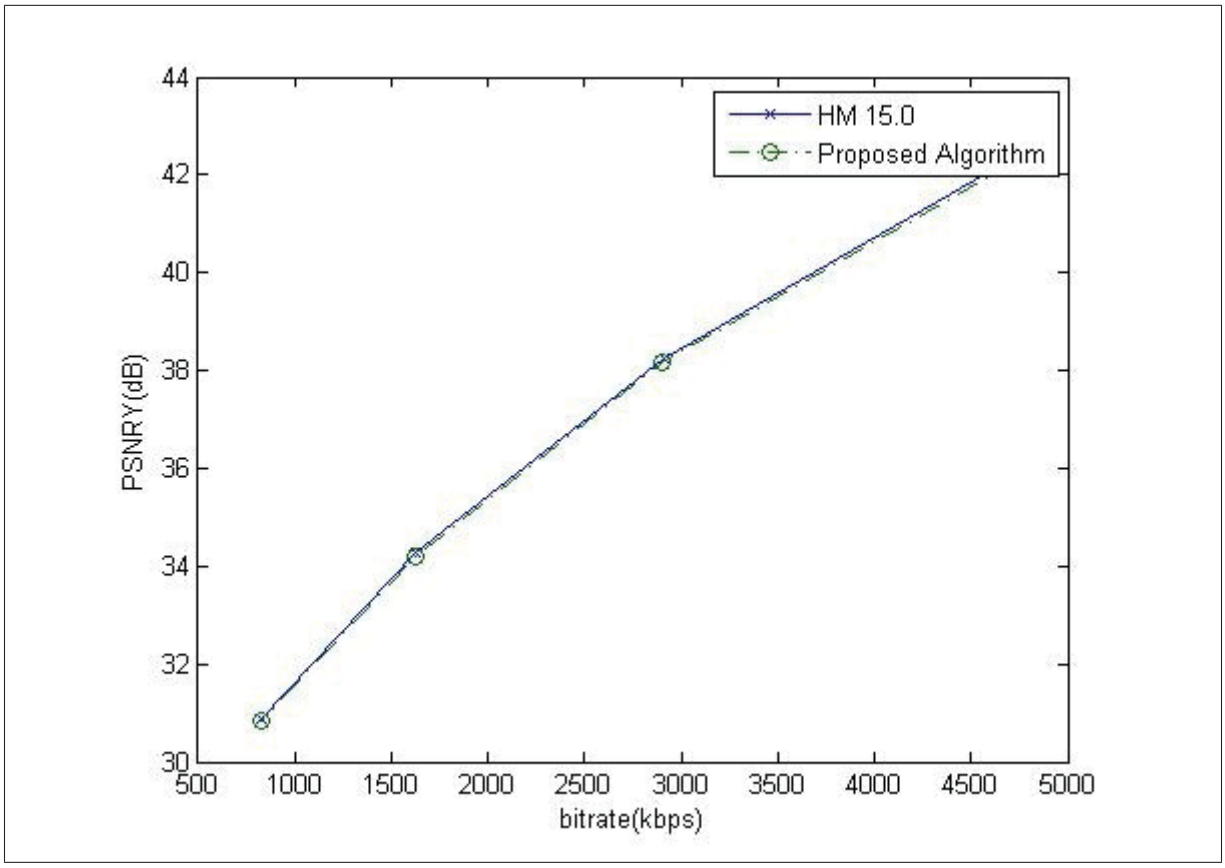

Figure 4.11 RD curve of the proposed method and HM, *RaceHorses*.

4.5 Experimental Results and Discussion for Overall Mode Decision

In this section, we provide the results for our mode decision method which includes three steps of RDO cost prediction, chroma mode decision and edge detection based on gradient analysis. The results are obtained based on the system diagram shown in fig. 4.1. The implementation conditions and configuration are as mentioned in section 4.1.3. Table 4.14 shows the overall results of the proposed method for the first hundred frames of the recommended video sequences. In addition, Table 4.15 shows the overall results for all frames of the sequences to present the results under common test conditions (CTC) recommended in Bossen (2013). We provide the results for all recommended sequences, using Prewitt as our selected gradient operator and method E for fast chroma mode decision. Overall, the proposed method provides a 47.3% time reduction with a very low BD-Rate increase of 1.37%. As mentioned, parameters can be tuned to obtain a different trade-off between time reduction and BD-Rate. Figures 4.12 and 4.13 show the rate-distortion curves of the proposed method versus HM for the BasketballPass and PartyScene sequences. Based on these figures, the two curves are very close, which shows that the proposed method leads to negligible quality loss, and is effective for different bitrates. Similar curves are achieved for other sequences with different video textures.

To compare our work to other methods, Table 4.16 provides results of state-of-the-art works on mode decision. To have fair comparisons with the reference papers, this table includes averages of different sets of sequences. Thus, based on these different sets of sequences, we have different test conditions as follows:

A) Test condition includes sequences shown in Table 4.15 other than ParkScene, BQMall, BasketballPass and Johnny, which are training sequences in (Zhang *et al.*, 2017a) and are not used to compute the average. In addition, we exclude RaceHorses as our training sequence. Thus, this test condition includes 15 sequences. All frames are considered for this test condition.

Table 4.14 Experimental results of the overall proposed method compared to HM, first hundred frames

Class	Video sequence	TR (%)	BD-Rate (%)	BD-PSNR (dB)
A	Traffic	-49.1	1.44	-0.067
	PeopleOnStreet	-49.6	1.69	-0.082
	Nebuta	-48.0	0.68	-0.042
	SteamLocomotive	-49.1	0.60	-0.024
B	Cactus	-47.5	1.46	-0.048
	Kimono	-49.6	1.55	-0.050
	ParkScene	-47.3	1.00	-0.038
	BasketballDrive	-49.7	2.36	-0.060
	BQTerrace	-47.9	0.88	-0.042
C	BQMall	-46.6	1.26	-0.065
	PartyScene	-40.2	1.03	-0.068
	RaceHorsesC	-45.1	0.78	-0.044
	BasketballDrill	-48.2	0.88	-0.039
D	RaceHorses*	-46.0	1.20	-0.069
	BasketballPass	-47.4	1.66	-0.088
	BlowingBubbles	-43.7	1.05	-0.055
	BQSquare	-39.6	1.33	-0.096
E	FourPeople	-49.4	1.76	-0.090
	Johnny	-49.7	2.23	-0.085
	KristenAndSara	-49.3	2.21	-0.103
Average (with training sequence)		**-47.1**	**1.35**	**-0.063**
Average (without training sequence)		**-47.2**	**1.36**	**-0.062**

*indicates the training sequence

B) Test condition includes sequences shown in Table 4.15 other than Nebuta, SteamLocomotive, BasketballDrive, BQTerrace, RaceHorsesC and BlowingBubbles and also Vidyo3 is used instead of KristenAndSara to have the same sequences as (Gao *et al.*, 2015). In addition, we exclude RaceHorses as our training sequence. Thus, this test condition includes 13 sequences. First hundred frames are considered for this test condition.

C) Test condition includes sequences shown in Table 4.15 (20 common test sequences as those considered in (BenHajyoussef *et al.*, 2017)). It should be noted that because this reference does not provide separate results for different test sequences for their mode decision approach and it is not possible to remove RaceHorses for comparison, we

Table 4.15 Experimental results of the overall proposed method compared to HM (CTC conditions)

Class	Video sequence (frames)	TR (%)	BD-Rate (%)	BD-PSNR (dB)
A	Traffic (150)	-48.8	1.46	-0.068
	PeopleOnStreet (150)	-49.4	1.71	-0.084
	Nebuta (300)	-49.2	0.68	-0.041
	SteamLocomotive (300)	-48.9	0.70	-0.026
B	Cactus (500)	-47.7	1.46	-0.048
	Kimono (240)	-49.5	1.54	-0.051
	ParkScene (240)	-47.4	1.02	-0.040
	BasketballDrive (500)	-49.1	2.37	-0.061
	BQTerrace (600)	-46.7	0.82	-0.035
C	BQMall (600)	-47.0	1.48	-0.073
	PartyScene (500)	-41.1	1.02	-0.068
	RaceHorsesC (300)	-44.6	0.65	-0.035
	BasketballDrill (500)	-48.7	0.85	-0.039
D	RaceHorses* (300)	-46.5	1.22	-0.065
	BasketballPass (500)	-46.8	1.71	-0.094
	BlowingBubbles (500)	-44.2	1.03	-0.061
	BQSquare (600)	-41.0	1.29	-0.085
E	FourPeople (600)	-48.9	1.78	-0.091
	Johnny (600)	-49.9	2.22	-0.085
	KristenAndSara (600)	-49.5	2.21	-0.102
Average (with training sequence)		**-47.2**	**1.36**	**-0.063**
Average (without training sequence)		**-47.3**	**1.37**	**-0.062**

*indicates the training sequence

compare our results including this sequence to their results to have exactly the same sequences as they use for computing the average. Considering the last two rows of Table 4.15, the difference between the results including or excluding RaceHorses is negligible. All frames are considered for this test condition.

D) Test condition includes sequences shown in Table 4.15 other than BasketballDrive, BQMall and BasketballPass, which are not used in (Tariq *et al.*, 2016). Similar to test condition C and since they do not provide separate results for different sequences (in their Top 1 Mode method), we include RaceHorses for comparison. Thus, this test condition includes 17 sequences. All frames are considered for this test condition.

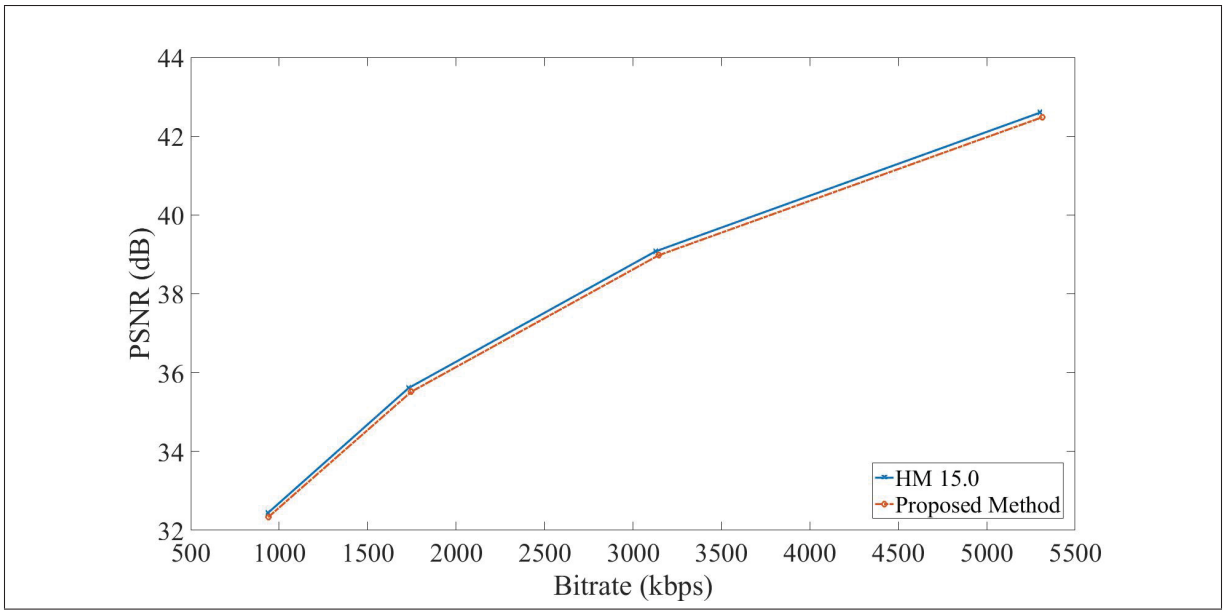

Figure 4.12 RD curves of the proposed mode decision method and HM, *BasketballPass*.

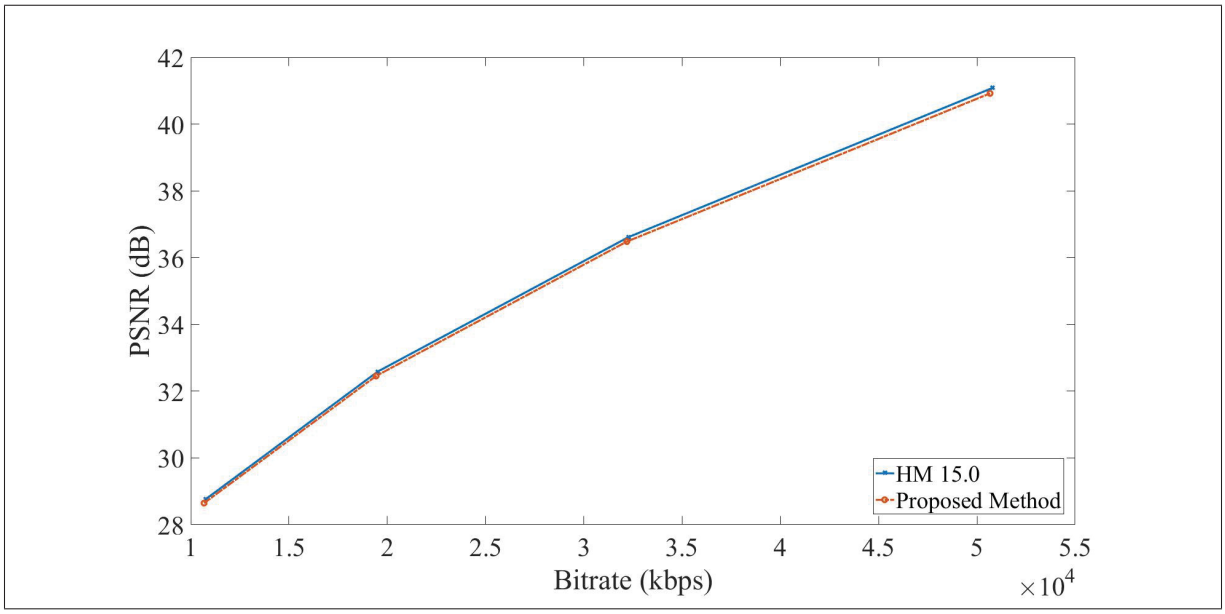

Figure 4.13 RD curves of the proposed mode decision method and HM, *PartyScene*.

Table 4.17 shows the results of the proposed method for extra sequences of class E which are not part of the common test conditions but are used to achieve some results in some works. They are presented here to permit a thorough comparison with other works as presented in

Table 4.16. In addition, Table 4.18 shows hit ratio of gradient analysis. It also shows the hit ratios of RDO cost modeling and chroma mode decision along with the combination of luma algorithms (gradient analysis and RDO cost modeling). The hit ratio is defined as the percentage of times that the best mode selected by the proposed algorithm is same as the mode selected by the HM.

Based on the comparison table, the proposed method offers a significantly higher time reduction compared to other state-of-the-art methods. It is nearly 10% faster than method presented in (Tariq *et al.*, 2016) with an improved BD-Rate. Methods presented in (BenHajyoussef *et al.*, 2017) and (Gao *et al.*, 2015) offer less than 0.5% BD-Rate improvement but are 15% and 20% slower respectively. Although method presented in (Zhang *et al.*, 2017a) offers a good BD-Rate, their time reduction is too small to be of much use in real-time systems. As it is increasingly difficult to maintain good BD-Rate as we increase the time reduction, the BD-Rate offered by the proposed method is quite impressive considering the time reduction of nearly 50%. In comparison to (Gao *et al.*, 2015), our method does not use the neighboring blocks as a reference for mode decision since such an approach could result in a domino effect and lead to degradation of the quality. Also, they use a method for reduction of RDO candidates which is based on a heuristic idea of dominant directions and have a limited time reduction. In contrast, our method is a solid algorithm based on various tests and statistical modeling. In (Tariq *et al.*, 2016), the RDO cost is formulated as a function of SAD and QP. The function is quadratic and lets the encoder avoid performing RDO process by estimating RDO cost. While they describe the RDO cost as a deterministic function of SAD, we consider RDO cost as a stochastic model which is described in the form of probability distributions. Authors in (Zhang *et al.*, 2017a) use a gradient-based approach for mode decision. In the same way, a gradient-based mode decision is proposed in (BenHajyoussef *et al.*, 2017). To improve this approach, they also propose an optimal mode selection based on the number of occurrences of each mode in the PU. In comparison to these two works, our method adds two significant extra steps of RDO cost prediction and chroma mode classification to achieve higher time reduction. It should be noted that there are other proposed methods in (BenHajyoussef *et al.*, 2017) and (Zhang *et al.*,

2017a) which are related to fast CU size decision and are not related to fast mode decision. It is important to note that the methods proposed in the current chapter can be combined with fast CU size decision methods in order to achieve even higher complexity reduction. To this end, in the next chapter, we propose our fast CU size decision method.

Table 4.16 Comparison of the proposed mode decision method with other MD methods

Test Condition	Methods	**TR (%)**	**BD-Rate (%)**
A	Proposed method	-47.2	1.30
	Zhang *et al.* (2017a)(MD)	-15.2	0.17
B	Proposed method	-47.3	1.44
	Gao *et al.* (2015)	-27.2	1.02
C	Proposed method	-47.2	1.36
	BenHajyoussef *et al.* (2017)(MD)	-31.8	0.9
D	Proposed method	-47.2	1.27
	Tariq *et al.* (2016)	-38.0	1.37

Table 4.17 Experimental results of the overall mode decision method compared to HM for class E sequences which are not part of CTC

Class	**Video sequence**	**TR (%)**	**BD-Rate (%)**	**BD-PSNR (dB)**
E	Vidyo1	-49.6	1.98	-0.087
	Vidyo3	-50.2	1.49	-0.071
	Vidyo4	-48.1	1.74	-0.070

Table 4.18 Hit ratio (%) for different algorithms, RaceHorses (RH), BasketballPass (BP), BlowingBubbles (BB)

		Hit ratio (%)				
QP		22	27	32	37	Average
RH	Gradient analysis (Prewitt)	71	73	75	77	74
	RDO cost modeling	74	78	79	82	78
	Chroma (method E)	73	79	83	87	80
	Luma algorithms combined	60	64	72	74	67
BP	Gradient analysis (Prewitt)	67	69	70	72	69
	RDO cost modeling	75	79	81	83	79
	Chroma (method E)	78	82	88	90	84
	Luma algorithms combined	57	60	69	70	64
BB	Gradient analysis (Prewitt)	72	74	77	79	76
	RDO cost modeling	77	81	84	86	82
	Chroma (method E)	77	82	87	90	84
	Luma algorithms combined	62	67	74	77	70

Summary

In this chapter, we proposed some novel methods for complexity reduction in HEVC intra coding mode decision. To this end, we added new modules to the encoder. First of all, an RDO cost modeling predicts the RDO costs based on the low-complexity RMD cost using a statistical method. We use goodness of fit tests to verify that RDO costs come from normal models which makes it possible to estimate the RDO costs based on RMD cost. In addition, RMD process is replaced by a gradient analysis step to select the most promising candidates. Using this method, irrelevant modes are excluded from further processing by the high complexity rate-distortion optimization. To this end, a gradient-based edge detection is proposed to determine the main directions in a block and exclude irrelevant angular modes from further processing. A gradient-based approach is useful since intra coding uses objects' edges to find the best mode (direction) for forming the prediction block. At this step, we compared four different gradient operators together where Prewitt showed slightly better results compared to other kernels. Then, to exploit the spatial correlation, we added highly likely neighboring modes to increase the accuracy of mode decision. In the next step, RDO dodging removes most modes which go through RDO process. In chroma mode decision, we proposed a mode classification

to put the prediction modes into two groups of promising and non-promising candidates. RDO process is eliminated for non-promising candidate modes which leads to high time reduction and almost no quality loss in chroma mode decision. In summary, implementing all mode decision methods, we got a 47.3% time reduction with a negligible quality loss of 1.37% of BD-Rate.

CHAPTER 5

PROPOSED CODING UNIT SIZE DECISION METHODS

As mentioned in chapter 2, HEVC uses a quadtree structure based on the CTU to efficiently encode video sequences with various kinds of visual textures. The CTU is the root of the tree and is considered as the largest CU. It is set to 64×64 by default in the HM. Each CU, including the CTU, is split recursively into four equal size CUs. This adaptive splitting approach is especially useful for higher resolution videos ($4K \times 2K(4K), 8K \times 4K(8K)$, etc.). The depth of the 64×64 CU is 0 and the maximum depth is 3, i.e., the size of the smallest CU is 8×8. To perform the prediction with intra coding, for each CU a PU is associated, except for the 8×8 CU, which could be predicted as an 8×8 PU or four 4×4 PUs. In CU splitting, there is an optimal tree for each block which leads to the best coding performance. However, finding such tree requires an exhaustive search which results in a long encoding time. An example of such tree is depicted in fig. 5.1. There are two general approaches to reduce the CU splitting complexity and have a fast CU size decision. The first approach is early splitting which aims at skipping mode decision at the current level and moving to the next depth without doing the RDO computations at the current depth. The second approach is early splitting termination which aims at terminating the CU splitting process and skipping mode decision at the next levels. In this chapter, we are going to propose some methods to efficiently implement these two approaches in the HEVC encoder.

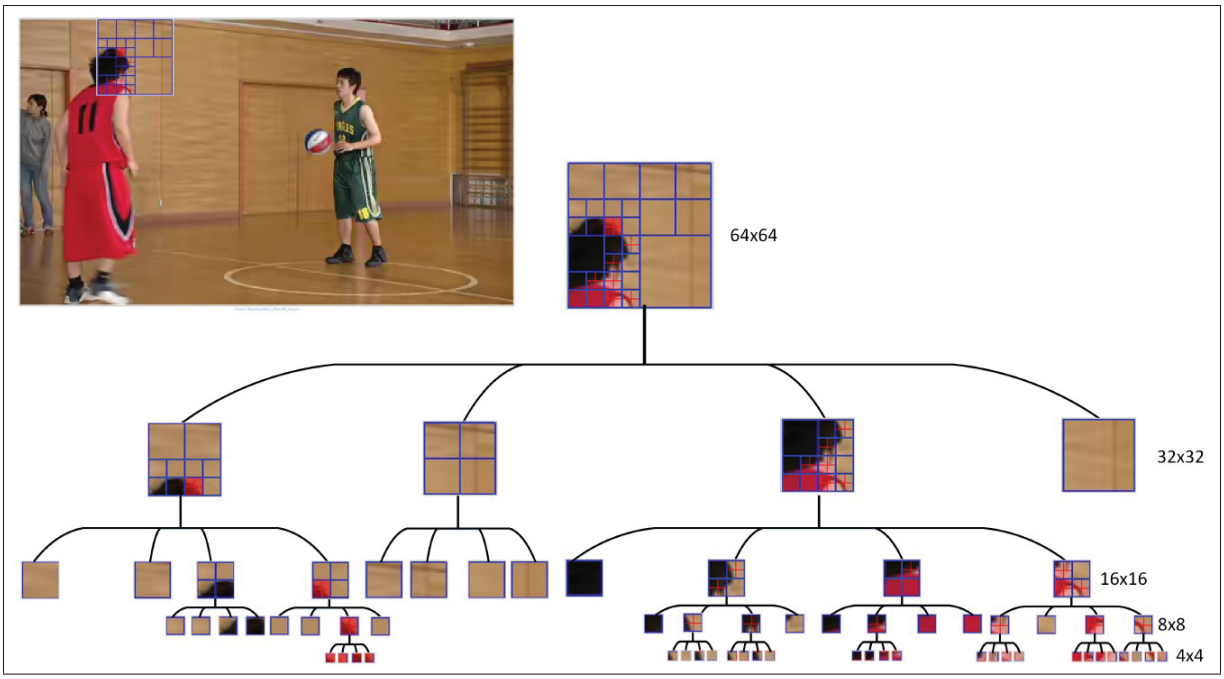

Figure 5.1 An optimal tree for efficient CU size decision (Shan, 2016).

5.1 Early Splitting Termination Based on Global and Directional Gradients

To lower the complexity associated with CU size decision in intra coding, in this section, we develop a method based on global and directional gradients to early terminate the CU splitting procedure and prevent processing of unnecessary depths. The global and directional gradients determine if the unit is predicted with high accuracy at the current level, and when that's the case, the CU is deemed to be non-split. In addition, this section shows the relevance of two gradient-based features which will be used in our general learning framework presented in section 5.3.

The general idea of CU early splitting termination is determining the splitting decision of the current CU by means that are less complex than performing the RDO process for the next depths. This is a binary classification problem in which $C = \{C_S, C_N\}$ is the set of classes. C_S and C_N are the classes of split and non-split CUs, respectively. A classification problem is solved based on a feature vector $F = \{f_1, f_2, f_3, \ldots\}$, in which the features need to be relevant and independent. Selecting relevant features can highly improve the CU classification process.

The relevance between features and classes can be assessed using mutual information (MI), defined as follows for two random variables F and C:

$$MI(F;C) = \int\int p(f,c) log \frac{p(f,c)}{p(f)p(c)} df dc, \tag{5.1}$$

where $p(f,c)$ is the joint PDF of F and C. MI shows the shared information between F and C, i.e., it represents the amount of uncertainty that is reduced for C when F is known. Mathematically, the MI of n features as random variables and the selected class is expressed as (Martinez-Enriquez *et al.*, 2011):

$$\begin{aligned} MI(F_1,F_2,\dots F_n,C) &= h(F_1,F_2,\dots F_n) - h(F_1,F_2,\dots F_n|C) \\ &= \int\int p_{\mathbf{f}c}(f_1,f_2,\dots f_n,c) log \frac{p_{\mathbf{f}c}(f_1,f_2,\dots f_n,c)}{p_{\mathbf{f}}(f_1,f_2,\dots f_n)p_c(c)} d\mathbf{f} dc, \end{aligned} \tag{5.2}$$

where $p_{\mathbf{f}}(f_1,f_2,\dots f_n)$, $p_c(c)$ and $p_{\mathbf{f}c}(f_1,f_2,\dots f_n,c)$ are the joint PDF of the n features, the PDF of the selected class and the joint PDF of the features and the selected class, respectively. The function h shows the Shannon entropy. Since the encoder performs the intra prediction by using the neighbouring pixels in different directions, the features which are based on the CU's edges and CU's gradient increase the mutual information between features and classes. Based on our extensive experiments with different video sequences, we select two gradient-based features, which result in an excellent trade-off between computational complexity and coding efficiency and are discussed in the next sub-sections.

5.1.1 Early Splitting Termination by Global Gradient

When a low degree of pixel variation is present at the current level, it is highly likely that the CU is not split into four smaller CUs to perform prediction. The global gradient is an appropriate measure for evaluating this variation. The mean of gradient amplitudes (MGA) is given by:

$$MGA = \frac{1}{n} \sum_i \sum_j |G_X(i,j)| + |G_Y(i,j)|, \tag{5.3}$$

where the sum is over all n pixels across the CU. To compute the gradient components G_X and G_Y at each pixel, the Sobel operator is applied as a simple and low-complexity edge detector with 3×3 convolution masks. The sum of the absolute values of the vertical and horizontal components is used as an approximation of the gradient amplitude to avoid a resource-demanding square root operation. The CUs with larger *MGA* tend to split while those with smaller *MGA* are predicted at the current level. Figure 5.2 shows the discrimination ability of this feature. It should be noted that when the QP is higher, the CU selects larger sizes as an optimum solution. From various experiments, we could observe that in the *MGA* graph, the effect of QP to discriminate split and non-split regions performed well when assumed linear. In view of this, to decide whether the CU splitting at the current level can be terminated, a measure is proposed as:

$$f_1 = \frac{MGA}{\alpha} - QP, \tag{5.4}$$

where α is a coefficient assigned for each block size. The CU is of the non-split class if $f_1 < Th_1$ where the CU splitting process is terminated. Otherwise, we check whether the CU could be effectively predicted by an angular mode which is the subject of the next sub-section.

5.1.2 Early Splitting Termination by Directional Gradient

If the global gradient of the CU is large, the CU cannot be predicted by the dc or planar modes at the current depth, but it may still be predicted by an angular mode provided that the directional gradient along the mode is small. To evaluate this gradient, the mean of directional gradient amplitudes (MDGA) is given as:

$$MDGA = \frac{1}{n}\sum_i\sum_j(|G_X(i,j)| + |G_Y(i,j)|) \times cos(\theta(i,j)), \tag{5.5}$$

where, again, the sum is over all n pixels across the CU, G_X and G_Y are gradient components and θ is the angle between the gradient at each pixel and the best angular mode of the current CU. CUs with large *MGA* but small *MDGA* along the best angular mode can be effectively

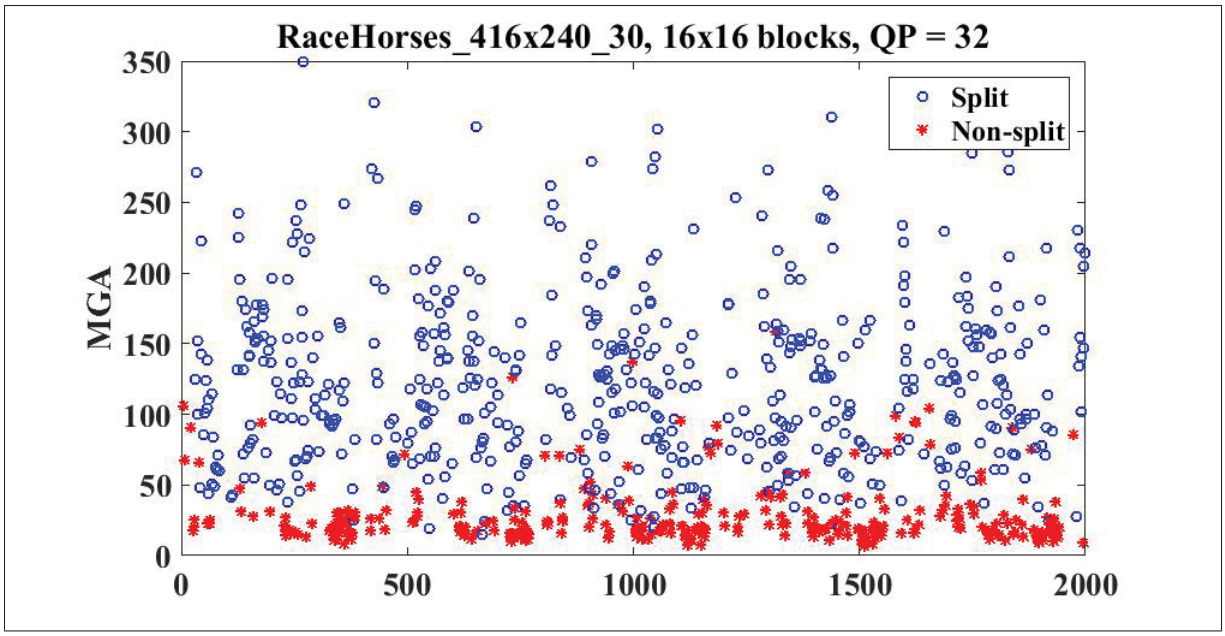

Figure 5.2 Split and non-split CUs using RDO and their MGA feature.

predicted at the current level. Figure 5.3 shows the discrimination ability of the *MDGA*. To use the *MDGA* as a tool for classifying the CU, we need to first determine the best angular mode of the CU. To that end, we utilize our low-complexity intra mode decision approach described in section 4.4. Once the best angular mode is known, the *MDGA* could be computed along this mode. Considering a linear effect of QP, as well as the block size, we have the following measure as the second feature:

$$f_2 = \frac{MDGA}{\beta} - QP, \tag{5.6}$$

where β is a coefficient assigned for each block size. The CU is of the non-split class if $f_2 < Th_2$, where the CU splitting process is terminated. Otherwise, the CU is split and the entire algorithm is repeated for the four newborn CUs.

5.1.3 Experimental Results and Discussion

The proposed early splitting termination is implemented in the HM using a PC equipped with an Intel Core^{TM} i7-4790 CPU @ 3.60 GHz and 32 GB of RAM. Test sequences recommended

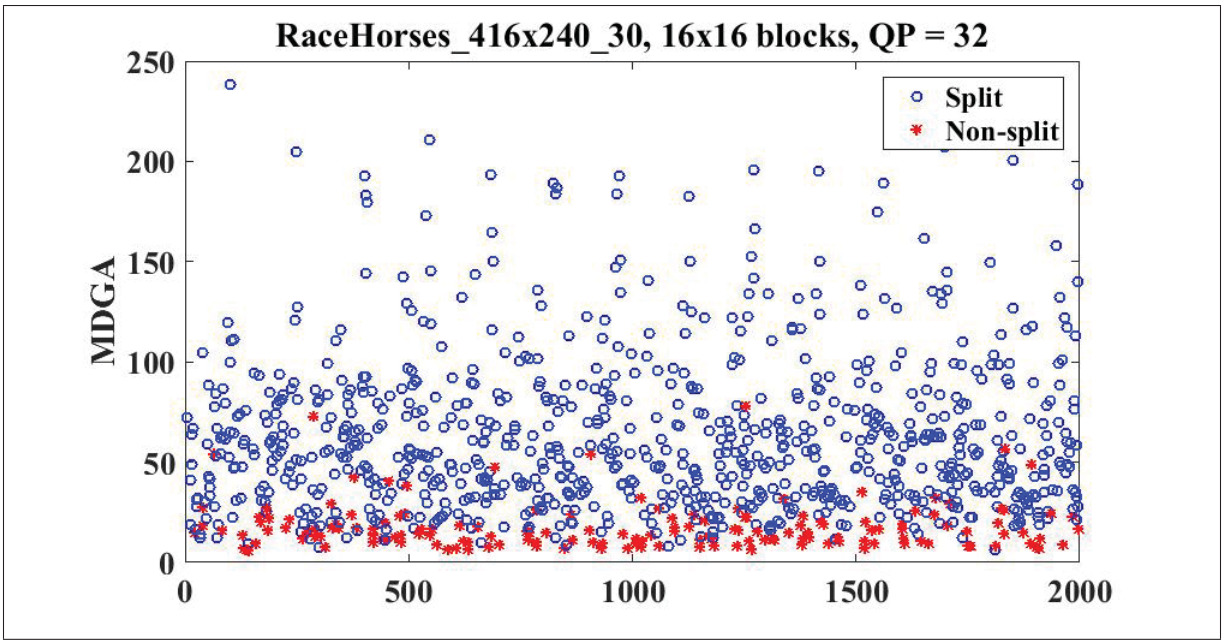

Figure 5.3 Split and non-split CUs using RDO and their MDGA feature.

in (Bossen, 2013) are used to conduct the experiments for the all-intra configuration and the Main profile. Parameter α is set to 1, 0.9, 0.4 and 0.3 for block sizes of 8×8, 16×16, 32×32 and 64×64, respectively, and parameter β is set to 0.8, 0.7, 0.2 and 0.1 for the same block sizes. We select the smaller parameters for larger block sizes since the larger blocks tend to split more than the smaller blocks. $Th1$ and $Th2$ are set to -5 and 0, respectively. By changing these thresholds, the proposed algorithm can provide an interesting trade-off between computational complexity and coding efficiency, which makes it a very flexible method for different applications. We obtain the results by combining the mode decision approach described in section 4.4 and the CU size decision method presented in this section. BD-Rate and BD-PSNR are used to compare our proposed method with the anchor HM by setting QP to 22, 27, 32 and 37. The results are presented in Table 5.1 as the time reduction (TR), BD-Rate and BD-PSNR compared to HM. To obtain the parameters, in our experiments, we used the *RaceHorses* sequence as a training sequence, and as a result, this sequence is excluded from the average results. As the table shows, our proposed method provides a 52%

time reduction, with a 0.07 dB quality loss which is a very good trade-off for the negligible quality loss.

Table 5.1 Experimental results of the early splitting termination method compared to HM, first hundred frames

Class	Video sequence	TR (%)	BD-Rate (%)	BD-PSNR (dB)
A	Traffic	-52.5	1.30	-0.061
	PeopleOnStreet	-47.8	1.29	-0.063
B	Cactus	-50.8	1.49	-0.049
	Kimono	-59.1	0.85	-0.028
	ParkScene	-49.7	0.67	-0.026
	BasketballDrive	-62.2	2.35	-0.059
	BQTerrace	-50.1	0.87	-0.041
C	BQMall	-49.6	1.75	-0.091
	PartyScene	-39.2	1.14	-0.076
	RaceHorsesC	-47.2	0.76	-0.043
	BasketballDrill	-48.4	1.39	-0.063
D	RaceHorses	-41.7	1.15	-0.065
	BasketballPass	-54.1	2.07	-0.110
	BlowingBubbles	-40.5	1.00	-0.052
	BQSquare	-39.2	1.41	-0.103
E	Vidyo1	-63.8	2.07	-0.091
	Vidyo3	-63.8	2.94	-0.141
	Vidyo4	-64.3	2.19	-0.088
Average		**-51.9**	**1.50**	**0.070**

5.2 Early Splitting and Early Splitting Termination Based on Bayesian Analysis

5.2.1 Proposed Method

In this section, we present our method for CU classification which uses two distortion-based costs to classify a block into split and non-split classes. The first distortion is between the original block and the predicted block and the second one is between the original block and the reconstructed block. To this end, we select a classification approach based on Bayesian classification because of its speed, low complexity and good accuracy for this problem.

Bayesian classifiers let us do the classification phase as fast as possible which is the most important measure to have an efficient and fast video encoder.

As mentioned before, to avoid the exhaustive search when finding the optimal CU splitting tree, we need early splitting to prevent doing computations at the current level and early split termination to terminate the process for the next levels. To apply these concepts of early splitting and early splitting termination in the encoder, we define two binary classification problems. Figure 5.4 shows the framework for these two problems. At the first step, based on some low complexity features the block is decided to be split or not. For splitting blocks, the computations at the current level are avoided and we go to the next level of the tree. For the other class, some more complex and accurate features are extracted and based on them the block is decided to be non-split so the splitting process is terminated and the encoder skips the next levels of the tree, otherwise the standard approach of the encoder is applied.

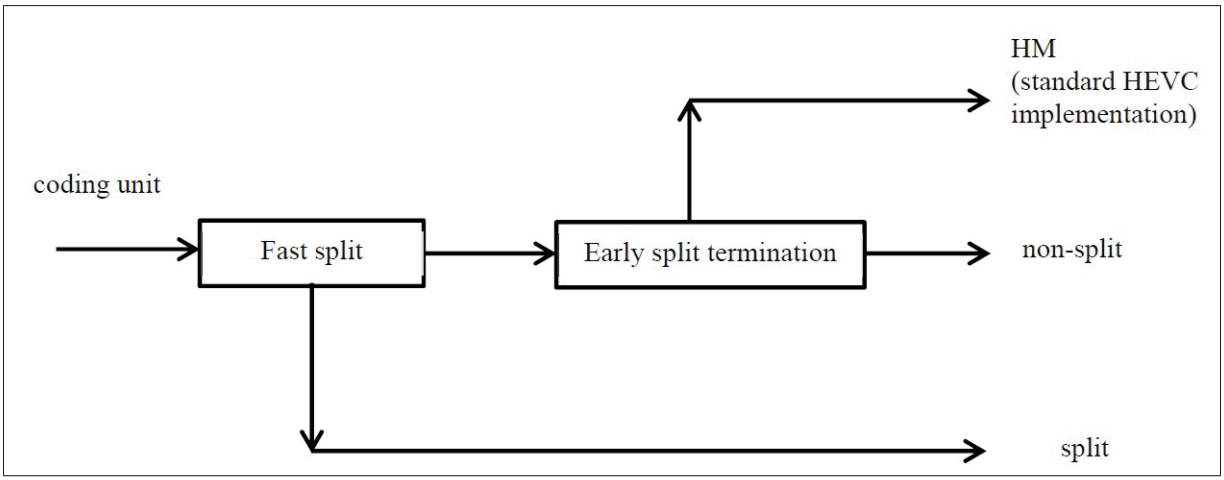

Figure 5.4 Block diagram of the two binary classification problems.

Similar to section 5.1, $C = \{C_S, C_N\}$ is the set of the classes while C_S and C_N are the classes of split and non-split CUs, respectively and the classification algorithm uses the feature vector $F = \{f_1, f_2, f_3, \ldots\}$ to classify the coding units. The probability that a specific block belongs to the class C_i is shown by $P(C_i|F), i \in \{S, N\}$. In Bayesian terminology, $P(C_i|F)$ is regarded as the a posteriori probability for two classes.

In a classification problem, making a wrong decision results in misclassification loss. For a Bayesian problem this loss is indicated by Bayesian risk. For this loss, we consider SSE between the block under coding and the reconstructed block as the loss for the early splitting termination problem and the bit rate for the fast split problem. We consider $L_{i,j}$ as loss of making decision as class C_i while the right class is C_j. In the case of right decision there is no loss, so $L_{i,i} = 0$. Based on these losses, the conditional Bayesian risk is computed as follows:

$$\begin{aligned} BR_C(C_N|F) &= L_{N,N}P(C_N|F) + L_{N,S}P(C_S|F) = L_{N,S}P(C_S|F) \\ BR_C(C_S|F) &= L_{S,S}P(C_S|F) + L_{S,N}P(C_N|F) = L_{S,N}P(C_N|F) \end{aligned} \tag{5.7}$$

If $BR_C(C_S|F) < BR_C(C_N|F)$ the split class is chosen and otherwise the choice is the non-split class. The a posteriori probabilities are computed based on the following Bayes' rule:

$$P(C_i|F) = \frac{p(F|C_i)P(C_i)}{p(F)}, i \in \{S,N\}, \tag{5.8}$$

where $P(C_i)$ is the a priori probability and $p(F|C_i)$ is a conditional PDF and they are computed offline during the training phase. Thus, the decision function, based on minimization of Bayesian risk, is given as follows:

$$\varphi(F) = \begin{cases} C_N & \text{if} \quad \frac{p(F|C_S)}{p(F|C_N)} < \frac{L_{S,N}P(C_N)}{L_{N,S}P(C_S)} \\ C_S & \text{elsewhere} \end{cases} \tag{5.9}$$

The Bayesian decision threshold on the right side of this equation is computed based on coding unit size and the QP.

The selected features for these Bayesian problems are RMD cost and RDO cost. Each of these two features are used for one of the mentioned binary classification problems to have one dimensional feature vectors which could be easily modeled by well-known distributions. Based on our experiments with different video sequences, these two features are selected because they result in a very good trade-off between computational complexity and coding performance. As

mentioned in chapter 2, RMD cost is based on SATD distortion between the block to code and the predicted block, a lagrange multiplier and the number of bits for signaling the best intra mode and is defined as follows:

$$f_1 = D_{SATD} + \lambda_1 \times B_{Mode}. \tag{5.10}$$

The second feature, RDO cost, is based on the SSE between the block to code and the reconstructed block, a lagrange multiplier and the number of bits for the encoded block after entropy coding and is defined as follows:

$$f_2 = D_{SSE} + \lambda_2 \times B_{Coding}. \tag{5.11}$$

The lagrange multipliers in these two equations are computed using the QP. f_1 is used for the first classification problem as a low complex feature. f_2 is more complex and more accurate and is used for the second problem of early splitting termination. We need to note that if, based on the first feature, the block is decided to be split, there is no need to compute the second feature. To show that the proposed features have the ability of discrimination, we have run the HEVC standard encoder for some video sequences with different QPs and different block sizes. Among them the results for *BlowingBubbles* (416×240) with QP equal to 22 and the coding unit size of 8×8 are reported. This is considered as the training phase for the Bayesian classifier which collects statistics for classes and features. Figures 5.5 and 5.6 show these features histograms for split and non-split blocks.

Based on these figures, the proposed features have high ability for block classification while involving low complexity to the encoder. The shown histograms are modeled as Gaussian distributions and the model parameters like mean and variance are extracted in the training phase. Then these parameters are used in the coding phase to classify the coding block based on the extracted features. The overall classification algorithm is summarized in algorithm 5.1. As it can be seen there is a recursive function that calls itself if the block needs to be split.

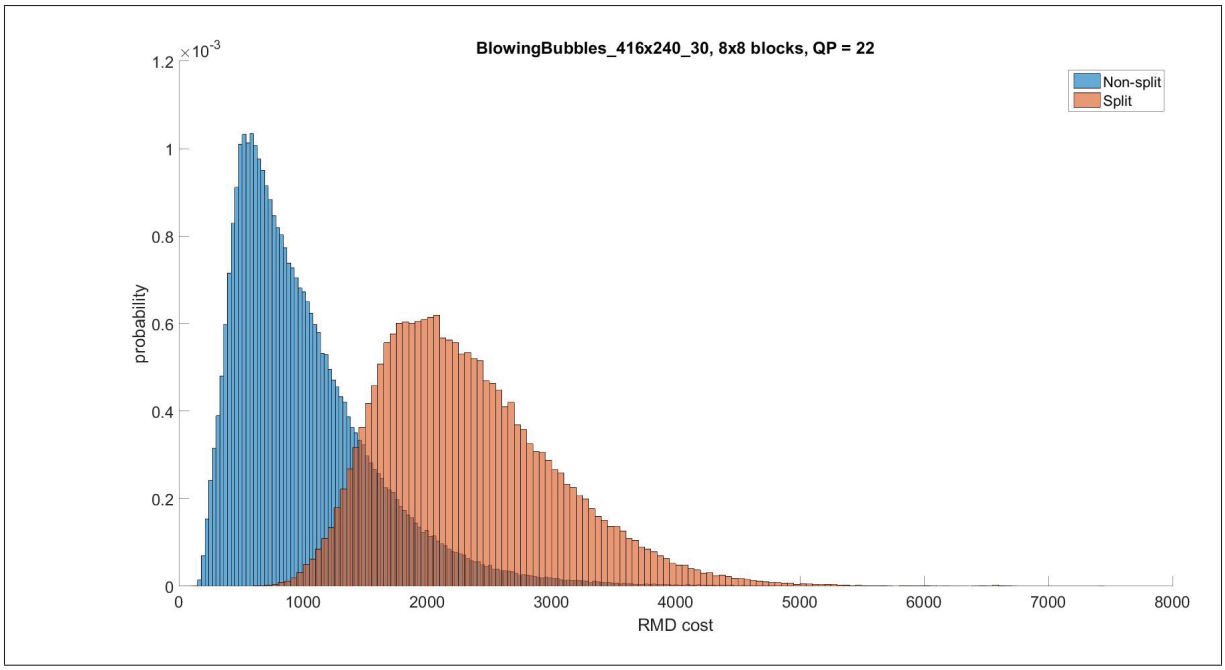

Figure 5.5 Histograms of the first feature for split and non-split blocks.

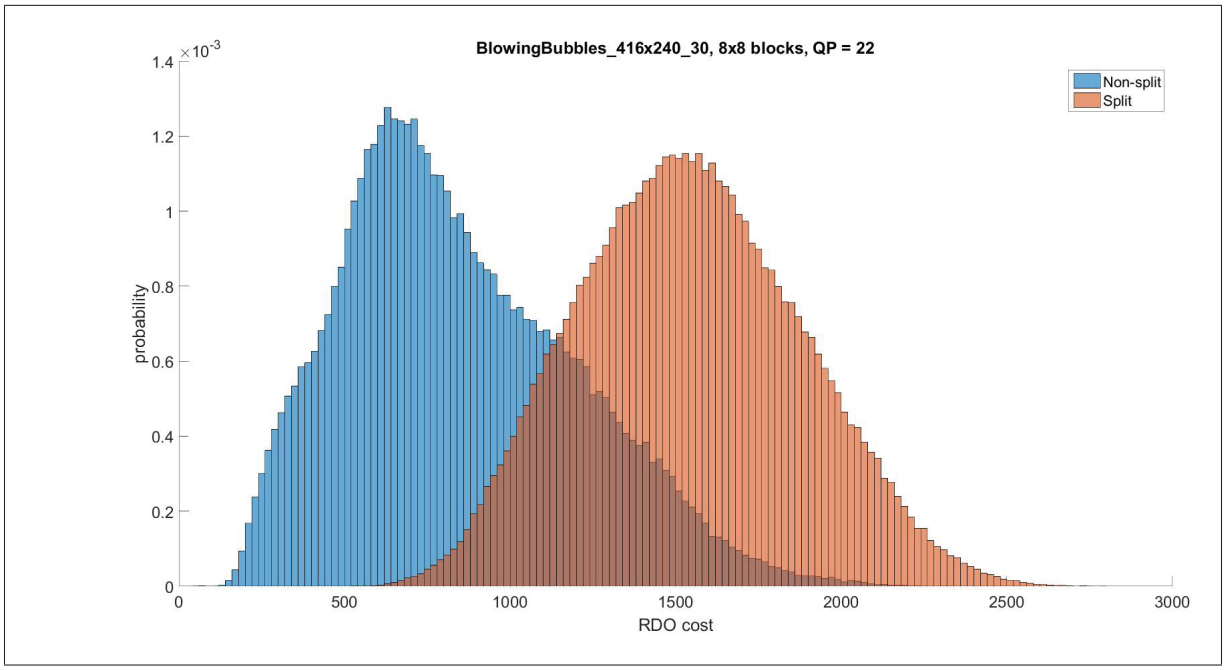

Figure 5.6 Histograms of the second feature for split and non-split blocks.

Algorithm 5.1 Classifying a coding unit into split or non-split

Input: *Thresholds for* φ *based on QP and block sizes*
Output: *Suboptimal splitting tree*

Start the process for largest block with depth=0 and size=64×64*, depthMax=4*
function *codingUnitSplit(CodingUnit, depth)*
  if $depth == depthMax$ **then**
    return "best mode" and "RDO cost" based on mode decision
  else
    Compute feature f_1
    if $\varphi_{fastsplit}(f_1) == C_N$ **then**
      Compute feature f_2
      if $\varphi_{earlytermination}(f_2) == C_N$ **then**
        return "best mode" and "RDO cost" based on mode decision (terminate splitting process)
      else if $\varphi_{earlytermination}(f_2) == C_S$ **then**
        run normal HM (standard HEVC implementation) for mode decision
        for $0 \leq i \leq 3$ **do**
          codingUnitSplit(sub_CodingUnit[i], depth+1)
        end for
      end if
    else if $\varphi_{fastsplit}(f_1) == C_S$ **then**
      for $0 \leq i \leq 3$ **do**
        codingUnitSplit(sub_CodingUnit[i], depth+1)
      end for
    end if
  end if
end function

5.2.2 Experimental Results and Discussion

In this section, we report results based on time reduction (TR), BD-PSNR and BD-Rate to compare the method described above to HM. *BlowingBubbles* sequence is used as the training sequence and is not used at the coding phase. The training is done for each QP and each block size and the data gathered during training phase is used for coding with the same QP and block size. The implementation platform is an Intel i7-3770 CPU-3.40, 12 GB of RAM, running Windows 7. We use recommended sequences in (Bossen, 2013) to implement the proposed method. HM is configured in All-Intra mode, and run for QPs of 22, 27, 32 and 37. Table 5.2 shows the results of these experiments in comparison to HM. The time reduction is

computed for each QP, and the average over all QPs is presented in the table. According to the experimental results, we achieve an average of 43.2% time reduction over all video sequences. The results for the training sequence are not included in the average. The main contributors to this time reduction are avoiding doing rate distortion optimization for the split blocks and terminating the process for the next levels of the coding tree for non-split blocks. The quality loss is, on average, a 0.04 dB in BD-PSNR and a 1.07% increment in the BD-Rate, which only slightly affects the rate-distortion performance. To justify this, fig. 5.7 shows the RD curves of the algorithm versus HM for the *BasketballPass* sequence. From this figure, we can see that both implementations have almost the same rate-distortion performance.

Table 5.2 Experimental results of the early splitting and early splitting termination method compared to HM, first hundred frames

Class	**Video sequence**	**TR (%)**	**BD-Rate (%)**	**BD-PSNR (dB)**
A	Traffic	-47	1.28	-0.06
	PeopleOnStreet	-46	1.40	-0.06
B	Cactus	-41	1.01	-0.03
	Kimono	-54	1.88	-0.06
	ParkScene	-42	0.90	-0.03
	BasketballDrive	-50	1.49	-0.03
	BQTerrace	-37	0.54	-0.03
C	BQMall	-43	0.71	-0.04
	PartyScene	-26	0.32	-0.02
	RaceHorsesC	-39	0.66	-0.04
	BasketballDrill	-42	1.08	-0.04
D	RaceHorses	-33	0.81	-0.05
	BasketballPass	-42	0.78	-0.04
	BlowingBubbles	-32	0.37	-0.01
	BQSquare	-31	0.21	-0.02
E	Vidyo1	-55	1.81	-0.08
	Vidyo3	-52	1.97	-0.09
	Vidyo4	-55	1.57	-0.06
Average		**-43.2**	**1.07**	**0.04**

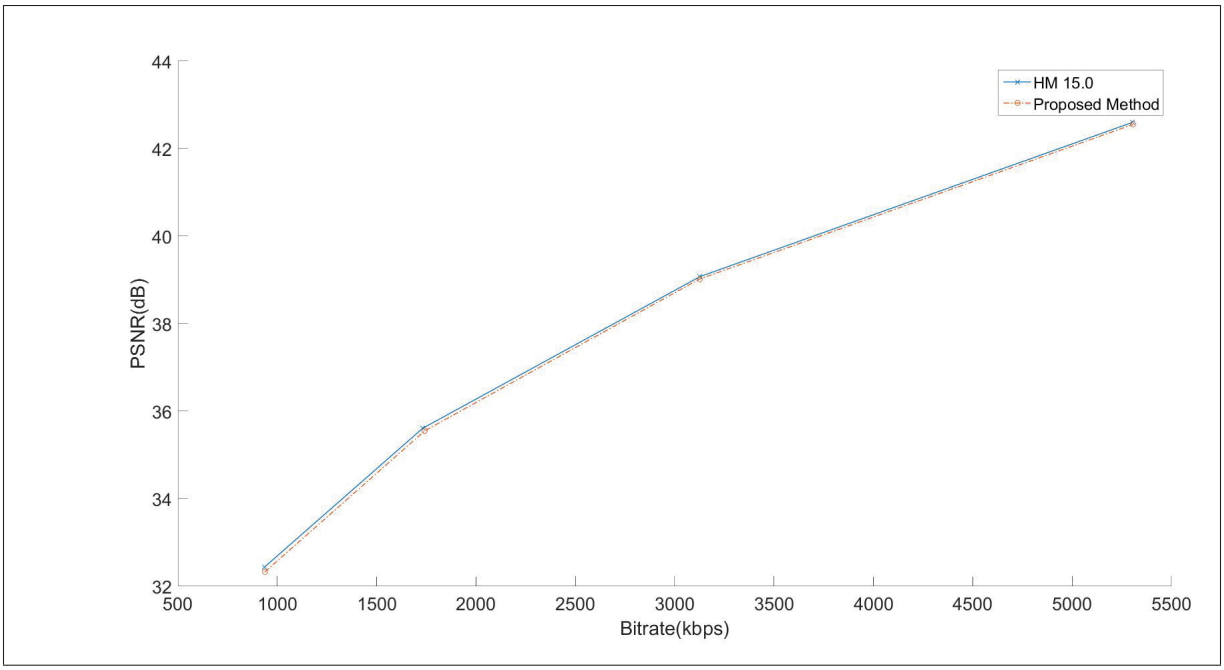

Figure 5.7 RD curve of the CU splitting method and HM, *BasketballPass*.

5.3 CU Size Decision Using Neural Network-Based Reinforcement Learning

In this section, we consider the CU size decision as a sequential decision making problem and formulate it by reinforcement learning (RL). RL is a set of powerful machine learning algorithms to find optimal solutions for complicated sequential problems. RL can be categorized somewhere between supervised learning and unsupervised learning. It is a mature field of study with more than fifty years of studies and in the last two decades, it has been highly related to the theory of Markov decision processes (MDPs). In RL, in contrast to programming where for each given state a proper action is determined a priori by the programmer, decisions are made by the interaction with the system, so it makes it easier to design a system by not requiring all the details in the design phase. It should be noted that the notations used in this section are adopted from (Wiering and van Otterlo, 2012).

In an RL system, an agent is learning how to take actions in an environment to minimize (maximize) a cost (reward) signal as a scalar feedback from the environment in the long run. The agent behaves in discrete time where at time t it observes the environment state

as s_t and then takes action a_t which results in immediate cost (reward) of $c_t(r_t)$ while the environment goes to the next state of s_{t+1}. Figure 5.8 shows a configuration of an RL system. The environment is modeled as a number of states and the agent takes actions from the set of actions to control the environment. The objective of such system is to take actions which result in maximization of a reward function or minimization of a cost function. To apply RL to our problem of CU splitting, we need to describe the problem as a sequential decision making problem where the CU is an MDP and the encoder is an agent sequentially taking coding decisions. RL algorithms intend to solve an MDP without prior knowledge about the system. The way that they approach the problem is by interaction and experiment with the environment which results in getting some kind of feedback in terms of reward or cost. On the other hand, if the dynamics of the environment and the cost function is known, dynamic programming (DP) methods are the solution for the problem.

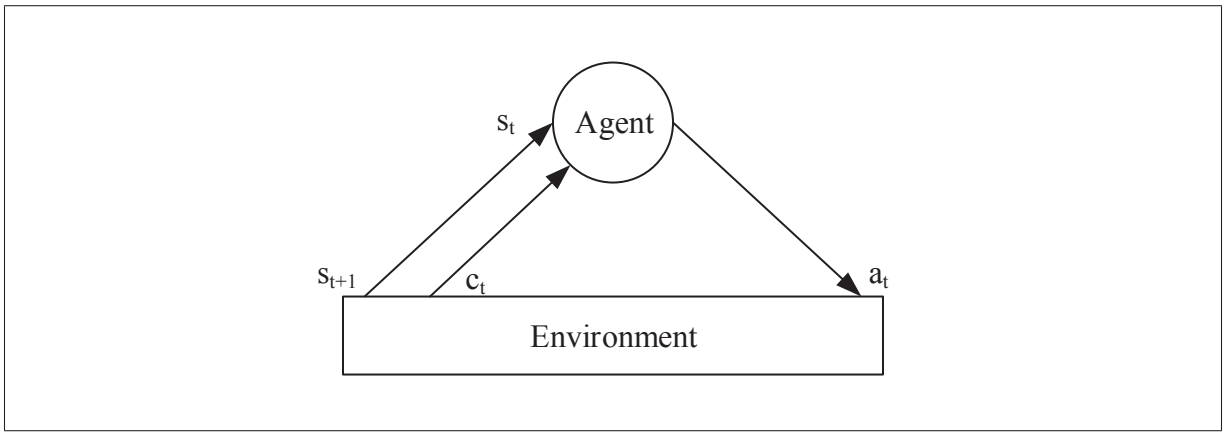

Figure 5.8 Reinforcement learning configuration (Sutton and Barto, 1998).

An MDP is defined as a tuple of (S,A,T,C) where S is a set of states, A is a set of actions, C is a stochastic cost function and T is a stochastic function which determines the transitions between the states. The sets of states and actions are defined as $S = \{s_1, s_2, ..., s_{N_S}\}$ and $A = \{a_1, a_2, ..., a_{N_A}\}$, respectively, where N_S is the number of states and N_A is the number of actions. A state could be described by some features or a pure state description. In general, not all actions are valid in all states. Usually, the agent has a limited number of actions in a given

state. By taking an action a, the agent changes the state of the MDP form s to s'. This transition between states is based on the transition function T which is defined as follows:

$$T : S \times A \times S \to [0,1], \tag{5.12}$$

which means if the agent takes action a in state s, the probability that the system goes to the next state s' is $T(s,a,s') = P(s'|s,a)$.

The cost function is another stochastic function which is defined as follows:

$$C : S \times A \times \mathbb{R} \to [0,\infty], \tag{5.13}$$

which means if the agent takes action a in state s, the probability density function of cost c is $C(s,a,c) = p(c|s,a)$. C as the immediate cost of an action is a real feedback signal. Defining the cost function is a challenging task in an RL system since it determines the goal of the learning. If we know the functions T and C, then the model of the MDP is known. The actions are taking place in a discrete order of time $t = 1,2,...$, so s_t and s_{t+1} are states at time t and $t+1$. The policy is a function which outputs an action a for a given state s of the MDP such as $a = \pi(s)$. A policy could be either deterministic which is defined as $\pi : S \to A$, or it could be stochastic which is defined as $\pi : S \times A \to [0,1]$. While the environment is modeled as MDP, the policy is considered as a part of the agent and is used to control the MDP. Policies also could be categorized as stationary where they are fixed or non-stationary where they change over time. The goal of the policy (agent) in episodic tasks is to minimize the following expected cost function:

$$E[\sum_{t=1}^{h} c_t]. \tag{5.14}$$

In episodic tasks, the agent behaves in some episodes, each of them starting from an initial state and ending in a terminal state (goal state) where the process terminates. In contrast, continuing tasks (infinite horizon tasks) run forever and the goal in these tasks is to minimize the following

expected cost function:

$$E[\sum_{t=1}^{\infty} \gamma^{t-1} c_t], \tag{5.15}$$

where $0 \leq \gamma < 1$ is a discount factor to make future costs less important than the immediate cost. Also, the discount factor makes sure that the expected cost is finite, even in the case of infinite horizon.

State-Value and Action-Value Functions

To know how good it is to be in a state or how good it is to take an action in a given state, value functions are defined for an MDP under a certain policy. The value of a state s while the agent is following policy π is shown by state-value function $V^{\pi}(s)$ and indicates the expected cost when starting from state s and then following policy π. This function is defined as follows (Wiering and van Otterlo, 2012):

$$V^{\pi}(s) = E_{\pi}\Big\{ \sum_{k=0}^{\infty} \gamma^{k} c_{t+k} | s_t = s \Big\}, \tag{5.16}$$

where E_{π} is the expected value of the long-term cost if the agent follows policy π. Similarly, the value of an action performed in a certain state is defined by state-action value function $Q^{\pi}(s,a)$ which indicates the expected return cost when starting from state s, taking action a and then following policy π. This function is called Q-function and defined as follows (Wiering and van Otterlo, 2012):

$$Q^{\pi}(s,a) = E_{\pi}\Big\{ \sum_{k=0}^{\infty} \gamma^{k} c_{t+k} | s_t = s, a_t = a \Big\}. \tag{5.17}$$

An MDP is solved when an optimal policy π^* is found for it, where $V^{\pi^*}(s) \leq V^{\pi}(s)$ for all policies π and all states s. Two general methods exist for solving MDPs. The first category is model-based methods and is based on DP and the second category is model-free which uses RL algorithms. In DP methods, the model of the MDP is known and it is used to compute the value functions and policies for the MDP. In RL methods, the model is not known and the agent tries to find the value functions and optimal policy of the MDP by interacting with the environment. In this category, samples of state transitions and immediate costs are used to

estimate the value functions. Since the model is not known, the environment has to be explored to gain information, and as a result the agent should deal with the exploration-exploitation trade-off.

Q-Learning (On-line Reinforcement Learning)

If the agent decides to update the estimates of the value functions after each interaction with the environment, the algorithm being used is in the category of online RL. Q-learning is a popular model-free method in this category which estimates the state-action value function (Q-function) (Watkins and Dayan, 1992). It is based on the idea of estimation of the Q-value of actions using the feedback in the form of cost and the current estimated Q-function. The update rule of Q-learning is as follows:

$$Q_{k+1}(s_t,a_t) = Q_k(s_t,a_t) + \alpha\Big(c_t + \gamma \min_a Q_k(s_{t+1},a) - Q_k(s_t,a_t)\Big), \tag{5.18}$$

where γ is the discount factor and $0 \leq \alpha \leq 1$ is the learning rate which determines the effect of new learned values in the estimation. In this method, the agent decides to take action a_t in the state s_t, then the environment gives cost c_t and goes to state s_{t+1} and based on this observation the Q-value of action a_t in the state s_t is updated. Q-learning can learn the optimal policy π^* regardless of the exploration policy which determines the selected action in each state. These kinds of algorithms called off-policy methods.

Function Approximation

If S and A are discrete sets with few members, it is possible to represent the value functions V and Q with the tabular form such as lookup tables. Otherwise, if the MDP is large with too many states and actions (or continuous states and actions) two problems emerge. First a huge memory is needed to store the value functions and second it would be very slow to learn the value functions for all states and actions. As a solution, and to avoid these two problems, value

functions are represented with function approximations as follows:

$$\hat{V}_\pi(s) \approx V_\pi(s), \tag{5.19}$$

$$\hat{Q}_\pi(s,a) \approx Q_\pi(s,a). \tag{5.20}$$

This makes it possible to generalize from observed states and actions to unobserved ones. Note that for the approximation to be valid the data related to observed states and actions (training data) should cover the range which is desired to be approximated. There are many function approximation methods such as linear algorithms, decision trees and neural networks. In our method to find a solution for CU size decision, we use neural networks as they are very capable of approximating continuous functions within specific range.

Batch-mode Reinforcement Learning

In batch-mode reinforcement learning, the agent tries to find the best policy using a set (batch) of transition samples. Batch-mode RL algorithms are popular mostly because of their two important features. First, they are data efficient and second they have a stable learning behavior. In batch-mode RL the agent receives a set of n transition samples $\mathscr{F} = \{(s_t, a_t, c_t, s_{t+1}) | t = 1, ..., n\}$ and then finds the best possible policy based on these samples. While in on-line learning the performance agent and the learning agent are the same, in batch-mode RL they are clearly separated with an additional agent for exploring. Here the learning agent is not interacting directly with the environment, but receives data in the form of four-tuples and extracts the best possible policy for the performance agent. Figure 5.9 shows these three agents of a batch-mode RL.

Generally, the explorer uses its own strategy and the learner has no idea how the explorer generates the training set. The samples could be collected by a purely random policy. The learner uses the training set to design a policy for the performer.

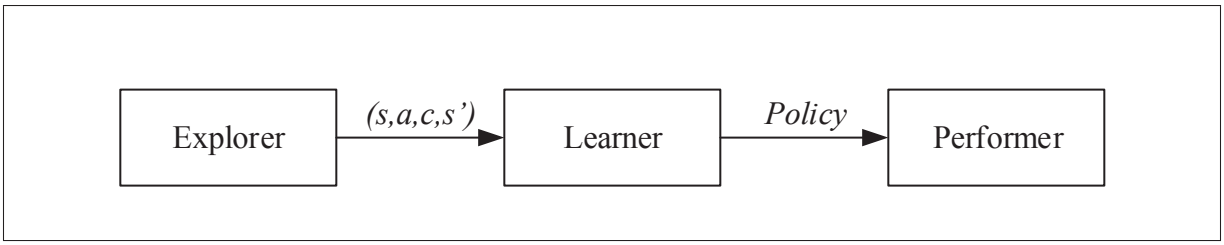

Figure 5.9 Three agents of a batch-mode reinforcement learning system.

Fitted Q Iteration

Fitted Q iteration (FQI) is a successful and popular batch RL algorithm proposed in 2005 (Ernst *et al.*, 2005). This algorithm combines Q-learning and batch-mode RL to take advantage of the benefits of both methods. It uses the n transition samples as the form of four-tuples (s,a,c,s') in the set $\mathscr{F} = \{(s,a,c,s')_l | l = 1,...,n\}$[1] to estimate the state-action value function (Q-function). Then the policy is derived based on this estimation. It starts by the estimation $\hat{Q}^0(s,a) = \bar{q}^0$ for all states and actions, where $\bar{q}^0$ is the initial value and is considered zero in (Ernst *et al.*, 2005). Then $\hat{Q}^1(s,a)$ is estimated using $\hat{Q}^0(s,a)$. Actually the exact $Q^1(s,a)$ is the conditional expectation of the immediate cost as follows:

$$Q^1(s,a) = E\{c_t | s_t = s, a_t = a\} \tag{5.21}$$

It means the expected cost of an action a in a given state s if the agent interacts with the environment just one time. After this first estimation the following two steps are iterated starting from $i = 0$ until a terminating condition is satisfied:

1- The set

$$\mathscr{T}^{i+1} = \{(s,a;\bar{q}^{i+1}_{s,a}) | \bar{q}^{i+1}_{s,a} = c + \gamma \min_{a' \in A} \hat{Q}^i(s',a')\} \tag{5.22}$$

is generated based on the all transition samples provided by set $\mathscr{F}$.

[1] $(s,a,c,s')_l = (s_l, a_l, c_l, s'_l)$

Algorithm 5.2 Fitted Q iteration

Input: $\mathscr{F} = \{(s,a,c,s')_l | l = 1,...,n\}$
Output: $\hat{Q}^N$ as an approximation of the value function Q^N
1 $k = 0$
2 initialize $\hat{Q}^0$ to zero everywhere on s and a
3 **repeat**
4 $\quad \mathscr{T} = \{(i_l, o_l), l = 1,...,n\}$ where:
5 $\quad i_l = (s_l, a_l), o_l = c_l + \gamma \min_{a' \in A} \hat{Q}^k(s'_l, a')$
6 $\quad$ derive $\hat{Q}^{k+1}$ from $\mathscr{T}$ using a supervised learning method
7 $\quad k = k + 1$
8 **until** $k = N$
9 **return** $\hat{Q}^N$

2- At this step, a supervised learning method is used to find $\hat{Q}^{i+1}$ as an approximation of Q^{i+1} based on the set $\mathscr{T}^{i+1}$.

Algorithm 5.2 shows all steps of the Fitted Q Iteration algorithm.

Having $\hat{Q}^N(s,a)$, and by assuming a discrete action space, obtaining the approximated optimal policy is straightforward as follows (Ernst *et al.*, 2005):

$$\hat{\pi}^*_N(s) = \operatorname*{argmin}_{a \in A} \hat{Q}^N(s,a), \tag{5.23}$$

and based on this policy the performance agent can take an action in a given state, although the state has never been seen before.

Neural Fitted Q Iteration

Neural Fitted Q Iteration (NFQ) is an efficient algorithm for approximation of the Q-function which uses multi-layer perceptron (MLP) (Riedmiller, 2005). Similar to FQI, it is based on training the value function using the stored experiences in the form of four-tuples. An acceptable policy can be derived, using the algorithm, by small number of interactions with the environment. NFQ is a batch-mode RL method based on FQI which uses neural networks as the supervised learning step to do the function approximation for value function. Neural networks

Algorithm 5.3 Neural fitted Q iteration

Input: $\mathscr{F} = \{(s,a,c,s')_l | l = 1,...,n\}$
Output: $\hat{Q}^N$ as an approximation of the value function Q^N
1 $k = 0$
2 initialize $\hat{Q}^0$ to zero everywhere on s and a
3 **repeat**
4 $\quad \mathscr{T} = \{(i_l, o_l), l = 1,...,n\}$ where:
5 $\quad i_l = (s_l, a_l), o_l = c_l + \gamma \min_{a' \in A} \hat{Q}^k(s'_l, a')$
6 $\quad$ Rprop$(\mathscr{T}) \rightarrow \hat{Q}^{k+1}$
7 $\quad k = k + 1$
8 **until** $k = N$
9 **return** $\hat{Q}^N$

are an excellent choice for this step as they are capable of approximating functions accurately and also because of their high degree of generalization from training set. Value functions are updated simultaneously using NFQ and the updates occur for all transition samples. As a result and in contrast to online RL, previous updates are not damaged by the new ones. As another advantage of NFQ and other fitted algorithms, one can apply easily a supervised learning algorithm because the updates take place simultaneously for all training samples. In NFQ, as a supervised learning algorithm, Rprop (Riedmiller and Braun, 1993) is used to do the approximation. Algorithm 5.3 shows the main steps of NFQ.

5.3.1 Problem Formulation

In this section, we deploy a batch-mode RL to find an encoding policy for coding unit size decision in the context of intra HEVC coding. If the system dynamics and the cost function are available, based on DP theories an optimal policy π^* exists for the system. However, in our problem, we aim at finding a policy based on a finite set of observed sample transitions. Since theoretically it is not possible to find an optimal policy based on this finite set of samples, our objective is to find an approximation of such policy. Generally, in our method we consider S to be mutli-dimensional and continuous and A to be a finite set of discrete actions. Also, we assume model-free approach where we don't make assumptions on the dynamics and transition

probabilities of the system. So we keep the prior knowledge as low as possible where it is limited to the representation of the state as a feature vector. We discuss this state representation and the cost function of the system in separate sections (sections 5.3.2 and 5.3.3).

As the first step of the proposed algorithm, there is an exploration phase where the encoder runs and the data is collected. For the exploration policy, we choose a random policy where the actions are chosen randomly to see the impact of good and bad actions. This phase operates as the exploring agent in the batch-mode RL setting. An interesting point about fitted algorithms is that they do not make assumptions on how the set of sample transitions are achieved. This makes it easy to collect four-tuples in episodic task such as our problem as we can gather these transitions over multiple independent episodes. Then, we choose neural networks and in particular MLPs in our algorithm for approximating the Q-function because they are capable of approximating nonlinear function very well. This NN step, as the learning agent, is the second module of the algorithm. Finally, in the main phase the encoder behaves as the performing agent by using the learned Q-function and takes actions based on the policy obtained in the learning phase.

Algorithm 5.4 shows the steps of the proposed method for CU size decision in the form of a fitted learning algorithm. Although it is usual to use the policy described in eq. (5.23) to derive the action at each state, there is another approach which is easier for implementation for obtaining the approximated optimal policy. In this approach, the set $\mathscr{F}$ is split based on different actions resulting in multiple $\mathscr{F}_a$ for each action. Then a regression algorithm can be applied to each of these subsets to derive the corresponding $\hat{Q}_a^N(s)$. The policy is then to compute all these Q-functions and select the action which its related $\hat{Q}_a^N(s)$ is the minimum. Based on this argument, in the algorithm, we apply the supervised learning (neural network) separately for each action (since we have discrete number of actions). After all iterations are done, a separate $\hat{Q}_a^N$ is achieved for each action. For a given state, the performance agent (encoder in our case) chooses the action with the lowest value of $\hat{Q}_a^N$ as the selected action. In the algorithm, N_A is the total number of actions and n_{a_i} is the number of four-tuples in $\mathscr{F}_{a_i}$. In addition, $A_{s'_l}$ is the set of valid actions in state s'_l.

Algorithm 5.4 Encoder training phase (learning agent) (inspired by (Gabel *et al.*, 2011))

Input: $\mathscr{F} = \{(s,a,c,s')_l | l = 1,...,n\}$
Output: $\hat{Q}_a^N$ as an approximation of the value function Q_a^N for all action $a \in A$

$k = 0$
initialize $\hat{Q}_{a_i}^0$ to zero everywhere on s, $i = 1$ to N_A
repeat
  for $i = 1$ to N_A **do**
    $\mathscr{F}_{a_i} = \{(s,a,c,s') \in \mathscr{F} | a = a_i\}$
    $\mathscr{T}_{a_i} = \{(i_l, o_l), l = 1,...,n_{a_i}\}$ where:
    $i_l = s_l, o_l = c_l + \gamma \min_{a' \in A_{s'_l}} \hat{Q}_{a'}^k(s'_l)$
    Function approximation based on NN
    $\mathrm{NN}(\mathscr{T}_{a_i}) \rightarrow \hat{Q}_{a_i}^{k+1}$
  end for
  $k = k + 1$
until $k = N$
return $\hat{Q}_{a_i}^N$, $i = 1$ to N_A

5.3.2 State Representation and Action Space

Although RL algorithms provide a solid framework to solve many complicated problems, they are still limited to low-dimensional state spaces. To apply RL methods to environments with high dimensionality such as raw images, the state space should be mapped to a low-dimensional feature space (Wiering and van Otterlo, 2012). In fact such mapping is part of the field knowledge applied by the system designer to solve the problem. It should be noted that the more prior field knowledge applied to a problem the more we are deviating from the basic idea of RL, so the mapping should be as wide as possible while we keep the dimensionality low. If the state space is n-dimensional the mapping is as follows:

$$\phi : R^n \rightarrow R^m, m \ll n \tag{5.24}$$

where m is the dimension of the feature space. As a result, the collected data set would be:

$$\mathscr{F}_\phi = \{(\phi(s), a, c, \phi(s')) | (s,a,c,s') \in \mathscr{F}\}, \tag{5.25}$$

in which $\phi(s) = (f_1(s), f_2(s), ..., f_{N_F}(s))$, where N_F is the number of features describing the state.

To find the best mapping ϕ, we use our experiments, described in previous sections of this thesis, to obtain the best state representation based on the most relevant features. In view of this, the features we have selected for the problem of CU size decision are as follows:

- *featureEarlySplitTermination*: shows whether the CU splitting is early terminated.

- *featureEarlySplitting*: shows whether the CU is early splitted.

- *featureStandardSplitting*: shows whether the CU is splitted similar to the standard HEVC implementation.

- *featureRMDcostBlock*: the RMD cost of the CU.

- *featureRMDcostBlockComputed*: shows whether the RMD cost is computed for the CU.

- *featureRDOcostBlock*: the RDO cost of the CU.

- *featureRDOcostBlockComputed*: shows whether the RDO cost is computed for the CU.

- *featureMGA*: the MGA of the CU as described in section 5.1.

- *featureMDGA*: the MDGA of the CU as described in section 5.1.

Using these features, we describe each state by a 9-dimensional feature vector. All the states are categorized into two classes. Terminal states and non-terminal states. Terminal states are those states where the encoding process is terminated for the CU and the episode ends. Terminal states are three categories as follows:

- Early terminated (ET) state, is the state of a CU for which the encoding process is terminated and the encoding of the lower depths is avoided. The *featureEarlySplitTermination* for this state is 1.

- Early split (ES) state, is the state of a CU for which the encoding process at the current depth is interrupted and the encoder moves to the lower depth. The *featureEarlySplitting* for this state is 1.

- Standard split (SS) state, is the state of a CU for which the encoding process at the current depth is fully performed and the encoder moves to the lower depth to continue the encoding process for the CTU. This CU is treated exactly as it is treated in the HM (the exhaustive approach). The *featureStandardSplitting* for this state is 1.

Non-terminal states are those for which *featureEarlySplitTermination*, *featureEarlySplitting* and *featureStandardSplitting* are zero and the encoding process of the corresponding CU has not yet terminated.

Action Space

The action space $A = \{a_1, a_2, ..., a_{N_A}\}$ contains N_A discrete number of actions. In our problem, there are in total five actions available for the encoder as the performance agent. Not all actions are valid in all states. We define two kinds of action in our problem. First, there are information-gathering actions (IGAs) which need to be taken just one time and gain some knowledge about the state of the system. For the purpose of this work, we consider obtaining the value of a feature as the desired knowledge. Second there are episode terminating actions (ETAs) which lead to a terminal state and like IGAs are taken just one time in each episode.

The reason to define and apply IGAs is well understood by considering the problem of active feature acquisition (Shim *et al.*, 2017) in a prediction scenario. Expensive features, in terms of computation, which are not much useful for an accurate prediction, should not be computed. In view of this, the agent should somehow know that is it worth to compute a given feature which increases the accuracy of prediction to some extent. This is a very important problem when we are dealing with systems with features which need a lot of computation resources to be computed. This problem is not usually considered in a traditional machine learning framework where the features are assumed to be known a priori and without any cost (feature acquisition

and prediction are assumed to be independent). To address this issue, we define IGAs so the agent associates the cost of computing a feature with the cost of taking an action in the RL context. This way, we incorporate the concept of active feature acquisition in our learning problem. Following is the list of actions along with the states they are valid at:

- *actEarlySplitTermination*

This action early terminates the splitting procedure and removes the lower depths to be processed. It is an ETA and it is valid when *featureRDOcostBlockComputed=true*.

- *actEarlySplitting*

This action early splits the CU into four blocks before completing the encoding process for the current depth. It is and ETA and it is valid when *featureRDOcostBlockComputed=false*.

- *actStandardSplitting*

This action splits the CU, as standard procedure in HM, into four blocks. It is an ETA and it is valid when *featureRDOcostBlockComputed=true*.

- *actComputeRMDcostBlock*

This action computes the RMD cost of the CU. It is an IGA and it is valid when *featureRMDcostBlockComputed=false*.

- *actComputeRDOcostBlock*

This action computes the RDO cost of the CU. It is an IGA and it is valid when *featureRDOcostBlockComputed=false* and *featureRMDcostBlockComputed=true*.

It should be noted that no action is valid in a terminal state because the episode ends in such state.

5.3.3 Cost Function

We define the cost function in a way which reflects the main trade-off of an encoding system between computational complexity and visual quality. Since we are aiming at designing a fast

encoder with high visual quality, the time required by each action as well as the quality loss in terms of rate-distortion by taking that action are considered as the costs for that action. The following definition provides a combination of these two costs:

$$c_a = c_{rd} + \eta c_t, \tag{5.26}$$

where c_{rd} is the rate-distortion cost and c_t is the computational time we pay as a result of taking an action. Since the rate-distortion cost itself is a combination of two terms, the exact definition of the cost function is as follows:

$$c_a = SSE + \lambda R + \eta c_t, \tag{5.27}$$

where *SSE* is the quality loss and R is the additional rate we need to pay while the encoder is deviating from optimal decisions made by an exhaustive search. To compute the c_t, we use the QueryPerformanceCounter function, which is a very useful Windows API to measure time intervals and acquire high-resolution time stamps. This function is used to get the current elapsed ticks, which could be used to measure time with a high resolution.

5.3.4 Experimental Results and Discussion

In this section, we present the results for the RL-based CU size decision. To obtain the results, we use the HEVC test model HM and a PC equipped with an Intel® Core™ i7-4790 CPU @ 3.60 GHz and 32 GB of RAM. The configuration and profile are set to *all-intra* and *Main profile*, respectively. The results are based on the common test conditions recommended in (Bossen, 2013), and are reported based on time reduction (TR), BD-Rate and BD-PSNR. The results are averaged over four QPs: 22, 27, 32 and 37. The training phase is done based on Algorithm 5.4. For Q-function approximation in line 9 of the algorithm, we use a multilayer perceptron neural network. This network is a fully connected one which comprises one input layer, two hidden layers and one output layer. The numbers of neurons in the first hidden layer

and second hidden layer are 18 and 9, respectively. The activation functions for the hidden layers are *tanh* and the activation function for the output layer is linear.

Table 5.3 shows the results for the proposed RL-based method. This method leads to a time reduction of 51.3% with a BD-Rate increase of 0.84%. This trade-off is quite interesting. In the exploring phase, the *RaceHorses* sequence is used as the training sequence, and hence, the average results are presented with and without considering this sequence. To compare our work to other methods, Table 5.4 and fig. 5.10 provide results of state-of-the-art works on coding unit size decision.

Table 5.3 Experimental results while implementing RL-based CU size decision compared to HM

Class	Video sequence	TR (%)	BD-Rate (%)	BD-PSNR (dB)
A	Traffic	-54.5	1.21	-0.059
	PeopleOnStreet	-54.4	1.25	-0.060
	Nebuta	-52.9	0.21	-0.018
	SteamLocomotive	-52.8	0.22	-0.019
B	Cactus	-49.6	0.85	-0.029
	Kimono	-59.4	1.68	-0.056
	ParkScene	-50.2	0.76	-0.032
	BasketballDrive	-57.1	1.33	-0.033
	BQTerrace	-46.9	0.42	-0.028
C	BQMall	-51.8	0.61	-0.036
	PartyScene	-37.6	0.27	-0.021
	RaceHorsesC	-48.5	0.52	-0.038
	BasketballDrill	-50.3	0.90	-0.039
D	RaceHorses	-43.6	0.65	-0.049
	BasketballPass	-50.3	0.65	-0.042
	BlowingBubbles	-42.9	0.27	-0.012
	BQSquare	-42.7	0.19	-0.020
E	FourPeople	-61.1	1.61	-0.078
	Johnny	-58.3	1.73	-0.088
	KristenAndSara	-60.3	1.45	-0.063
Average (with training sequence)		**-51.3**	**0.84**	**-0.041**
Average (without training sequence)		**-51.7**	**0.85**	**-0.041**

Table 5.4 Comparison of the proposed RL-based method to other CU size decision methods

	TR (%)	**BD-Rate (%)**
Proposed method	-51.3	0.84
BenHajyoussef *et al.* (2017)	-31.0	0.70
Zhu *et al.* (2018)	-39.8	0.47
Lee and Jeong (2017)	-55.5	1.01
Zhang *et al.* (2017b)	-50.3	1.41

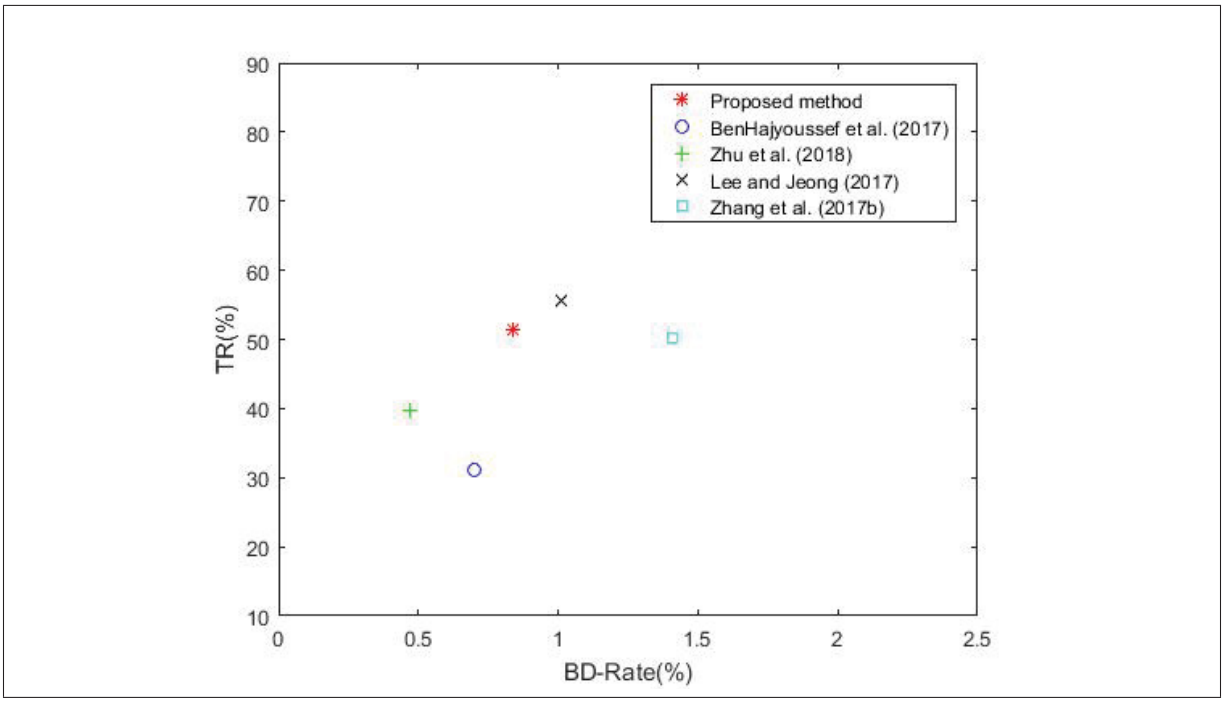

Figure 5.10 Comparison of the proposed RL-based method to other CU size decision methods.

We also report the results of the RL-based CU size decision when it is combined with the mode decision methods (RDO cost modeling and chroma mode decision) to present the overall time reduction we achieve in HEVC intra coding. Table 5.5 shows the results of this combination. Based on these results, we obtain a 62.4% time reduction for the intra coding which results in a very fast encoder. In addition, the quality loss is a 1.23% BD-Rate increment which negligibly affects the final visual quality. Here, again, the average results are presented with and without

RaceHorses as the training sequence. To compare our work to other methods, Table 5.6 and fig. 5.11 provide results of state-of-the-art works which combine coding unit size decision and mode decision.

Our method is specially useful for sequences where there are dominant modes for most CTUs. This is the case in sequences such as *SteamLocomotive* where there is one large object in a homogeneous background. In these sequences, the homogeneous areas are predicted by dc mode and the object's edge is predicted by directional modes. On the other hand, for sequences such as *PeopleOnStreet* where there are many small objects in a frame, CTUs may contain multiple objects. As a results there is no dominant mode for most CTUs and the exhaustive search performs better for these sequences. Our method also works well for sequences such as *BQTerrace* where there are long distinct edges in a frame which makes it easier to find the best modes as well as the proper CU sizes.

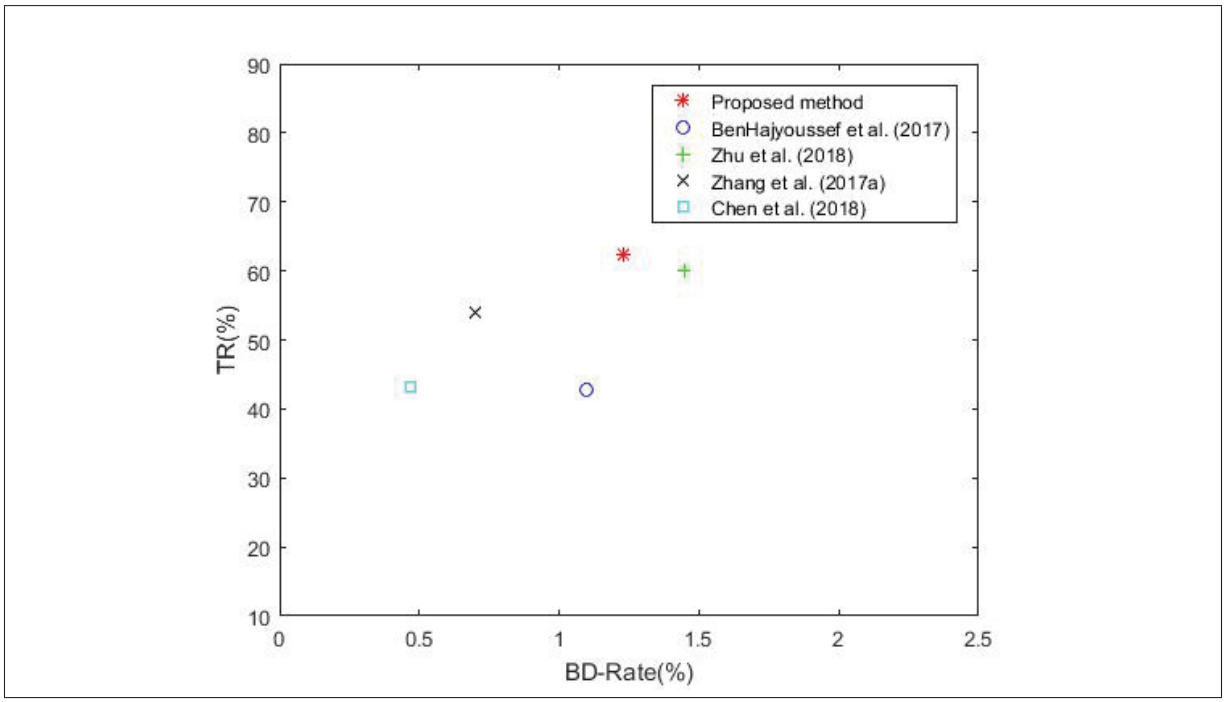

Figure 5.11 Comparison of the proposed RL-based CU size decision method combined with mode decision to other combined methods.

Table 5.5 Experimental results while implementing RL-based CU size decision combined with mode decision compared to HM

Class	Video sequence	TR (%)	BD-Rate (%)	BD-PSNR (dB)
A	Traffic	-64.7	1.75	-0.090
	PeopleOnStreet	-65.4	1.82	-0.094
	Nebuta	-64.4	0.38	-0.031
	SteamLocomotive	-63.6	0.40	-0.028
B	Cactus	-61.4	1.21	-0.044
	Kimono	-67.8	2.37	-0.084
	ParkScene	-61.5	1.15	-0.051
	BasketballDrive	-67.0	1.79	-0.047
	BQTerrace	-59.6	0.64	-0.042
C	BQMall	-62.9	0.96	-0.058
	PartyScene	-51.5	0.53	-0.042
	RaceHorsesC	-59.8	0.76	-0.054
	BasketballDrill	-62.5	1.19	-0.055
D	RaceHorses	-57.4	1.05	-0.076
	BasketballPass	-62.1	1.01	-0.065
	BlowingBubbles	-56.2	0.55	-0.030
	BQSquare	-54.8	0.47	-0.044
E	FourPeople	-69.1	2.17	-0.113
	Johnny	-67.1	2.35	-0.117
	KristenAndSara	-68.2	2.07	-0.098
Average (with training sequence)		**-62.4**	**1.23**	**-0.063**
Average (without training sequence)		**-62.6**	**1.24**	**-0.062**

Table 5.6 Comparison of the proposed RL-based CU size decision method combined with mode decision (MD) to other combined methods (CU size decision+MD)

	TR (%)	BD-Rate (%)
Proposed method	-62.4	1.23
BenHajyoussef *et al.* (2017)	-42.8	1.10
Zhu *et al.* (2018)	-60.0	1.45
Zhang *et al.* (2017a)	-53.9	0.70
Chen *et al.* (2018)	-43.2	0.47

Summary

In this chapter, we proposed novel methods based on two general approaches to reduce the CU splitting complexity and have a fast CU size decision. In the first approach, the CU is

early split and the mode decision computations are excluded from the current level. In the second approach, the CU splitting process is early terminated and all computations for the next levels are omitted which results in a significant time reduction. In CU split early termination, a method based on global and directional gradients is developed to prevent processing of unnecessary depths. The global and directional gradients determine if the unit is predicted with high accuracy at the current level, and when that's the case, the CU is deemed to be non-split. Then, we presented our method for CU classification based on Bayesian analysis which uses two distortion-based costs to classify a block into split and non-split classes. Bayesian classification is selected because of its speed, low complexity and good accuracy for this problem. In last section of the chapter, we considered the CU size decision as a sequential decision making problem and formulate it by reinforcement learning. To this end, we deployed a batch-mode RL to find an encoding policy for coding unit size decision in the context of intra HEVC coding. In our problem, we aim at finding a policy based on a finite set of observed sample transitions. Also, we proposed new features for CU size decision in this framework which are more relevant than features proposed in the literature. Combining this RL method with our proposed mode decision, we got a 62.6% time reduction with a small quality loss of 1.24% of BD-Rate.

CONCLUSION

In this thesis, we presented an overview of the HEVC intra coding followed by our proposed methods for encoding complexity reduction in two areas of mode decision and CU size decision. In addition, we reviewed the HEVC video coding standard from the computational complexity point of view and at the literature review we looked at some works that present solutions to alleviate the HEVC complexity. Compared to other work on HEVC intra coding complexity reduction, our contributions in this thesis are summarized as follows:

RDO cost modeling and prediction

By using this approach, we reduce the number of candidates in RDO process as much as possible. RDO cost prediction prevents high-demanding rate-distortion optimization for modes with no chance of being the best intra mode of the prediction unit. This is based on statistical modeling of the RDO cost using the low-complexity RMD cost.

Fast intra chroma mode decision

A chroma mode classification is proposed to avoid the RDO process for non-promising candidates.

Low complexity edge detection for mode decision

The rough mode decision of the HEVC test model is replaced by gradient analysis, based on the Prewitt operator, in order to decrease the number of candidates by excluding non-relevant directional modes. This algorithm improves the edge detection approaches by attributing three modes for each detected edge.

SATD cost classification, most relevant modes and RDO dodging

A binary classification, based on a dominant distance among SATD costs, and RDO dodging are proposed to reduce the number of candidate modes entering the RDO process. In addition, we propose to enhance the schemes

that exploit the spatial correlation among neighboring blocks by introducing the concept of most relevant modes.

CU splitting early termination

This method is based on global and directional gradients. The global gradient determines whether the CU is smooth enough to be predicted by dc or planar modes at the current depth. The directional gradient is used for non-smooth CUs and classifies a CU as a non-split CU when it could be predicted efficiently by an angular mode at the current level; thus, the CU splitting process is terminated. The proposed method benefits from some controllable parameters, which allow a flexible trade-off between the compression rate and the encoder complexity.

Early splitting and early splitting termination

In this approach, we use Bayesian classifiers to classify a CU into split and non-split based on two distortion-based features. The solution includes two binary classification problems for early splitting and early splitting termination.

CU size decision using reinforcement learning

An RL-based method is proposed to eliminate the non-necessary CU levels from RDO process. In this method, we use the concept of sequential feature acquisition to determine whether a feature is worth to be computed for the purpose of accurate classification.

Experimental results show that our contributions, combing mode decision methods and CU size decision algorithms, provide a 62.4% time reduction and a 1.23% BD-Rate increase which is a very solid trad-off between computational complexity and compression efficiency. Our proposed methods are based on adjustable parameters which can be tuned to get the desired compromise between computational complexity and coding efficiency; a characteristic which makes the proposed methods attractive for various applications.

As future work, we propose following research directions:

- There are only a few works on chroma complexity reduction. More research needs to be done on this area to design fast HEVC video encoders.
- We have used RL for CU size decision while it can also be used in mode decision to select the best candidate without running the RDO process. Also, in mode decision, generalized gaussian model can be used to model the empirical data instead of normal model.
- In RL framework for CU size decision, other supervised learning methods can be used for function approximation. Decision tree could be a choice in this area.
- Future research, in CU size decision, can introduce new features which are low in computational complexity and also relevant to predict the optimal CU size. In addition, promising classifiers such as convolutional neural networks can be used for both problems of CU size decision and mode decision.
- There is a lack of a comparison measure in the area of complexity reduction. As a future work, one can propose a metric that can fairly compare different methods considering the trade-off between coding efficiency and time saving. In the state-of-the-art work, the results are reported based on two metrics of BD-Rate and time reduction.
- And finally, the successful methods in HEVC complexity reduction which have been proposed in last years can be applied to 360 video and 3D applications to address new demands in these areas.

LIST OF REFERENCES

BenHajyoussef, A., T. Ezzedine, and A. Bouallègue. 2017. "Gradient-based pre-processing for intra prediction in High Efficiency Video Coding". *EURASIP Journal on Image and Video Processing*, vol. 2017, n° 1, p. 9.

Bhat, H. S. and N. Kumar. 2010. "On the derivation of the Bayesian Information Criterion". *School of Natural Sciences, University of California*.

Bjøntegaard, G. April 2001. "Calculation of average PSNR differences between RD-curves". *Video Coding Experts Group (VCEG) of ITU-T, VCEG-M33*.

Bossen, F. 2013. "Common test conditions and software reference configurations". *Joint Collaborative Team on Video Coding (JCT-VC), JCTVC-L1100*.

Bossen, F., B. Bross, K. Suhring, and D. Flynn. Dec 2012. "HEVC Complexity and Implementation Analysis". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, n° 12, p. 1685-1696.

Chen, F., D. Jin, Z. Peng, G. Jiang, M. Yu, and H. Chen. Apr 2018. "Fast intra coding algorithm for HEVC based on depth range prediction and mode reduction". *Multimedia Tools and Applications*.

Chen, G., L. Sun, Z. Liu, and T. Ikenaga. Nov 2013. "Fast mode and depth decision HEVC intra prediction based on edge detection and partitioning reconfiguration". In *2013 International Symposium on Intelligent Signal Processing and Communication Systems*. p. 38-41.

Cho, S. and M. Kim. Sept 2013. "Fast CU Splitting and Pruning for Suboptimal CU Partitioning in HEVC Intra Coding". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, n° 9, p. 1555-1564.

da Silva, T. L., L. V. Agostini, and L. A. da Silva Cruz. Aug 2012. "Fast HEVC intra prediction mode decision based on EDGE direction information". In *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*. p. 1214-1218.

Ernst, D., P. Geurts, and L. Wehenkel. 2005. "Tree-based batch mode reinforcement learning". *JOURNAL OF MACHINE LEARNING RESEARCH*, vol. 6, p. 503–556.

Fang, X., X. Zhu, L. Yu, and X. Shen. 2013. "Fast HEVC intra coding unit size decision based on an improved Bayesian classification framework". In *Picture Coding Symposium (PCS), 2013*. p. 273–276. IEEE.

Gabel, T., C. Lutz, and M. Riedmiller. April 2011. "Improved neural fitted Q iteration applied to a novel computer gaming and learning benchmark". In *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*. p. 279-286.

Gao, L., S. Dong, W. Wang, R. Wang, and W. Gao. 2015. "Fast intra mode decision algorithm based on refinement in HEVC". In *Circuits and Systems (ISCAS), 2015 IEEE International Symposium on*. p. 517–520. IEEE.

Helle, P., H. Schwarz, T. Wiegand, and K. R. Müller. May 2017. "Reinforcement learning for video encoder control in HEVC". In *2017 International Conference on Systems, Signals and Image Processing (IWSSIP)*. p. 1-5.

Hojati, E. 2018. "Massively Parallel Rate-Constrained Motion Estimation for High Efficiency Video Coder". Master's thesis, École de technologie supérieure. to be published.

Il-Koo, K. April 2014. "High Efficiency Video Coding (HEVC) Test Model 15 (HM15) Encoder Description". *Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11*.

Jamali, M. and S. Coulombe. May 2016a. "RDO cost modeling for low-complexity HEVC intra coding". In *2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*. p. 1-5.

Jamali, M. and S. Coulombe. Sept 2016b. "Coding Unit Splitting Early Termination for Fast HEVC Intra Coding Based on Global and Directional Gradients". In *2016 IEEE Workshop on Multimedia Signal Processing (MMSP)*.

Jamali, M. and S. Coulombe. 2018. "Fast HEVC Intra Mode Decision Based on RDO Cost Prediction". *Accepted in IEEE Transactions on Broadcasting*.

Jamali, M., S. Coulombe, and F. Caron. Oct 2014. "Method and system for fast mode decision for high efficiency video coding". https://patents.google.com/patent/US20160127725A1/en.

Jamali, M., S. Coulombe, and F. Caron. April 2015. "Fast HEVC Intra Mode Decision Based on Edge Detection and SATD Costs Classification". In *2015 Data Compression Conference*. p. 43-52.

Jiang, W., H. Ma, and Y. Chen. April 2012. "Gradient based fast mode decision algorithm for intra prediction in HEVC". In *2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)*. p. 1836-1840.

Kim, J., S. Jeong, S. Cho, and J. S. Choi. Jan 2012. "Adaptive Coding Unit early termination algorithm for HEVC". In *2012 IEEE International Conference on Consumer Electronics (ICCE)*. p. 261-262.

Kim, K. and W. W. Ro. 2018. "Fast CU Depth Decision for HEVC using Neural Networks". *IEEE Transactions on Circuits and Systems for Video Technology*, p. 1-1.

Korkmaz, S., D. Goksuluk, and G. Zararsiz. 2014. "MVN: an R package for assessing multivariate normality". *The R Journal*, vol. 6, n° 2, p. 151–162.

Lainema, J., F. Bossen, W. J. Han, J. Min, and K. Ugur. Dec 2012. "Intra Coding of the HEVC Standard". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, n° 12, p. 1792-1801.

Lee, D. and J. Jeong. 2017. "Fast intra coding unit decision for high efficiency video coding based on statistical information". *Signal Processing: Image Communication*, vol. 55, p. 121 - 129.

Li, S., S. Chen, J. Wang, and L. Yu. May 2009. "Second Order Prediction on H.264/AVC". In *2009 Picture Coding Symposium*. p. 1-4.

Liu, X., Y. Liu, P. Wang, C. F. Lai, and H. C. Chao. 2016. "An Adaptive Mode Decision Algorithm Based on Video Texture Characteristics for HEVC Intra Prediction". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. PP, n° 99, p. 1-1.

Liu, X., Y. Li, D. Liu, P. Wang, and L. T. Yang. 2017. "An Adaptive CU Size Decision Algorithm for HEVC Intra Prediction based on Complexity Classification using Machine Learning". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. PP, n° 99, p. 1-1.

Ma, S., S. Wang, S. Wang, L. Zhao, Q. Yu, and W. Gao. March 2013. "Low Complexity Rate Distortion Optimization for HEVC". In *2013 Data Compression Conference*. p. 73-82.

Martinez-Enriquez, E., A. Jimenez-Moreno, M. Angel-Pellon, and F. Diaz de Maria. 2011. "A Two-Level Classification-Based Approach to Inter Mode Decision in H.264/AVC". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, n° 11.

Marzuki, I., J. Ma, Y. Ahn, and D. Sim. 2016. "A context-adaptive fast intra coding algorithm of high-efficiency video coding (HEVC)". *Journal of Real-Time Image Processing*, p. 1–17.

Mecklin, C. J. and D. J. Mundfrom. 2005. "A Monte Carlo comparison of the Type I and Type II error rates of tests of multivariate normality". *Journal of Statistical Computation and Simulation*, vol. 75, n° 2, p. 93-107.

Min, B. and R. Cheung. 2015. "A fast CU size decision algorithm for the HEVC intra encoder". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, n° 5, p. 892–896.

Ohm, J. R., G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand. Dec 2012. "Comparison of the Coding Efficiency of Video Coding Standards-Including High Efficiency Video Coding (HEVC)". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, n° 12, p. 1669-1684.

Pakdaman, F., M. Hashemi, and M. Ghanbari. 2016. "Fast and efficient intra mode decision for HEVC, based on dual-tree complex wavelet". *Multimedia Tools and Applications*, p. 1–16.

Pan, F., X. Lin, S. Rahardja, K. Lim, Z. Li, D. Wu, and S. Wu. 2005. "Fast mode decision algorithm for intraprediction in H. 264/AVC video coding". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, n° 7, p. 813–822.

Park, M. and J. Jeong. 2015. "Fast HEVC Intra Prediction Algorithm with Enhanced Intra Mode Grouping Based on Edge Detection in Transform Domain". *Journal of Advances in Computer Networks*, vol. 3, n° 2, p. 1686–1694.

Richardson, I. E., 2010. *The H.264 advanced video compression standard*. ed. 2nd. West Sussex, United Kingdom : John Wiley & Sons Ltd, 316 p.

Riedmiller, M. 2005. "Neural Fitted Q Iteration - First Experiences with a Data Efficient Neural Reinforcement Learning Method". *ECML 2005, LNCS (LNAI), Springer, Heidelberg*, vol. 3720, p. 317–328.

Riedmiller, M. and H. Braun. 1993. "A direct adaptive method for faster backpropagation learning: the RPROP algorithm". In *IEEE International Conference on Neural Networks*. p. 586-591 vol.1.

Ruiz, D., G. Fernández-Escribano, J. Luis Martínez, and P. Cuenca. 2016. "Fast intra mode decision algorithm based on texture orientation detection in HEVC". *Signal Processing: Image Communication*, vol. 44, p. 12-28.

Ruiz-Coll, D., V. Adzic, G. Fernández-Escribano, H. Kalva, J. L. Martínez, and P. Cuenca. Oct 2014. "Fast partitioning algorithm for HEVC Intra frame coding using machine learning". In *2014 IEEE International Conference on Image Processing (ICIP)*. p. 4112-4116.

Saurty, K., P. C. Catherine, and K. M. S. Soyjaudah. Nov 2015. "Terminating CU splitting in HEVC intra prediction using the Hadamard Absolute Difference (HAD) cost". In *2015 SAI Intelligent Systems Conference (IntelliSys)*. p. 836-841.

Shan, Y. 2016. "Fast Intra-frame Coding Algorithm for HEVC Based on TCM and Machine Learning". Master's thesis. http://hdl.handle.net/10012/10917.

Shang, X., G. Wang, T. Fan, Y. Li, and Y. Zuo. 2016. "Low-Complexity Intra-Coding Scheme for HEVC". *Circuits, Systems, and Signal Processing*, vol. 35, n° 12, p. 4331–4349.

Shen, L., Z. Zhang, and Z. Liu. 2014. "Effective CU size decision for HEVC intracoding". *IEEE Transactions on Image Processing*, vol. 23, n° 10, p. 4232–4241.

Shen, X., L. Yu, and J. Chen. May 2012. "Fast coding unit size selection for HEVC based on Bayesian decision rule". In *2012 Picture Coding Symposium*. p. 453-456.

Shim, H., S. J. Hwang, and E. Yang. 2017. "Why Pay More When You Can Pay Less: A Joint Learning Framework for Active Feature Acquisition and Classification". *CoRR*, vol. abs/1709.05964.

Song, Y., Y. Zeng, X. Li, B. Cai, and G. Yang. 2017. "Fast CU size decision and mode decision algorithm for intra prediction in HEVC". *Multimedia Tools and Applications*, vol. 76, n° 2, p. 2001–2017.

Sullivan, G. J., J. R. Ohm, W. J. Han, and T. Wiegand. 2012. "Overview of the High Efficiency Video Coding (HEVC) Standard". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, n° 12, p. 1649–1668.

Sutton, R. and A. Barto, 1998. *Reinforcement learning: An introduction*. Adaptive computation and machine learning.

Sze, V., M. Budagavi, and G. J. Sullivan. 2014. "High efficiency video coding (HEVC)". *Integrated Circuit and Systems, Algorithms and Architectures. Springer*, p. 1–375.

Tariq, J., S. Kwong, and H. Yuan. 2016. "HEVC intra mode selection based on Rate Distortion (RD) cost and Sum of Absolute Difference (SAD)". *Journal of Visual Communication and Image Representation*, vol. 35, p. 112-119.

Trujillo-Ortiz, A., R. Hernandez-Walls, K. Barba-Rojo, and L. Cupul-Magana. 2007a. "HZmvntest:Henze-Zirkler's Multivariate Normality Test". *A MATLAB file. [WWW document]. URL http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=17931*.

Trujillo-Ortiz, A., R. Hernandez-Walls, K. Barba-Rojo, and L. Cupul-Magana. 2007b. "Roystest:Royston's Multivariate Normality Test". *A MATLAB file. [WWW document]. URL http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=17811*.

Vanne, J., M. Viitanen, T. D. Hamalainen, and A. Hallapuro. Dec 2012. "Comparative Rate-Distortion-Complexity Analysis of HEVC and AVC Video Codecs". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, n° 12, p. 1885-1898.

Wang, L. L. and W. C. Siu. Oct 2013. "Novel Adaptive Algorithm for Intra Prediction With Compromised Modes Skipping and Signaling Processes in HEVC". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, n° 10, p. 1686-1694.

Wang, Y., X. Fan, L. Zhao, S. Ma, D. Zhao, and W. Gao. Oct 2014. "A fast intra coding algorithm for HEVC". In *2014 IEEE International Conference on Image Processing (ICIP)*. p. 4117-4121.

Watkins, C. and P. Dayan. May 1992. "Q-learning". *Machine Learning*, vol. 8, n° 3, p. 279–292.

Wiegand, T., G. J. Sullivan, G. Bjøntegaard, and A. Luthra. 2003. "Overview of the H.264/AVC video coding standard". *IEEE Transactions on circuits and systems for video technology*, vol. 13, n° 7, p. 560–576.

Wiering, M. and M. van Otterlo, 2012. *Reinforcement Learning: State-of-the-Art*, volume 12 of *Adaptation, Learning, and Optimization*. Berlin, Germany : Springer.

Yan, S., L. Hong, W. He, and Q. Wang. Nov 2012. "Group-Based Fast Mode Decision Algorithm for Intra Prediction in HEVC". In *2012 Eighth International Conference on Signal Image Technology and Internet Based Systems*. p. 225-229.

Yazici, B. and S. Yolacan. 2007. "A comparison of various tests of normality". *Journal of Statistical Computation and Simulation*, vol. 77, n° 2, p. 175–183.

Yu, Q., X. Zhang, S. Wang, and S. Ma. Dec 2012. "Early termination of coding unit splitting for HEVC". In *Proceedings of The 2012 Asia Pacific Signal and Information Processing Association Annual Summit and Conference*. p. 1-4.

Zhang, H. and Z. Ma. 2013. "Early termination schemes for fast intra mode decision in high efficiency video coding". In *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on*. p. 45–48. IEEE.

Zhang, H. and Z. Ma. April 2014. "Fast Intra Mode Decision for High Efficiency Video Coding (HEVC)". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, n° 4, p. 660-668.

Zhang, M., C. Zhao, and J. Xu. Sept 2012. "An adaptive fast intra mode decision in HEVC". In *2012 19th IEEE International Conference on Image Processing*. p. 221-224.

Zhang, Q., X. Huang, X. Wang, and W. Zhang. 2015. "A Fast Intra Mode Decision Algorithm for HEVC Using Sobel Operator in Edge Detection". *International Journal of Multimedia and Ubiquitous Engineering*, vol. 10, n° 9, p. 80–90.

Zhang, T., M. T. Sun, D. Zhao, and W. Gao. Aug 2017a. "Fast Intra-Mode and CU Size Decision for HEVC". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, n° 8, p. 1714-1726.

Zhang, Y., H. Wang, and Z. Li. March 2013. "Fast Coding Unit Depth Decision Algorithm for Interframe Coding in HEVC". In *2013 Data Compression Conference*. p. 53-62.

Zhang, Y., Z. Pan, N. Li, X. Wang, G. Jiang, and S. Kwong. 2017b. "Effective Data Driven Coding Unit Size Decision Approaches for HEVC Intra Coding". *IEEE Transactions on Circuits and Systems for Video Technology*, p. 1-1.

Zhao, L., X. Fan, S. Ma, and D. Zhao. 2014. "Fast intra-encoding algorithm for high efficiency video coding". *Signal Processing: Image Communication*, vol. 29, n° 9, p. 935–944.

Zhao, W., L. Shen, Z. Cao, and Z. Zhang. 2012. Texture and correlation based fast intra prediction algorithm for HEVC. *Advances on Digital Television and Wireless Multimedia Communications*, p. 284–291. Springer.

Zhu, W., Y. Yi, H. Zhang, P. Chen, and H. Zhang. April 2018. "Fast mode decision algorithm for HEVC intra coding based on texture partition and direction". *Journal of Real-Time Image Processing*.

Ziou, D., S. Tabbone, et al. 1998. "Edge detection techniques-an overview". *Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii*, vol. 8, p. 537–559.